GNSS-SDR

0.0.22

Generated by Doxygen 1.18.0

1 GNSS-SDR 1

1.1 Contents . . . 1
1.2 Overview . . . 2
1.3 Building GNSS-SDR . . . 3
1.3.1 Debug and Release builds . . . 5
1.3.2 Updating GNSS-SDR . . . 5
1.4 Using GNSS-SDR . . . 5
1.5 Control plane . . . 6
1.5.1 Configuration . . . 6
1.5.2 GNSS block factory . . . 7
1.6 Signal Processing plane . . . 7
1.6.1 Signal Source . . . 8
1.6.2 Signal Conditioner . . . 8
1.6.3 Channel . . . 9
1.6.3.1 Acquisition . . . 9
1.6.3.2 Tracking . . . 10
1.6.3.3 Decoding of the navigation message . . . 11
1.6.4 Observables . . . 11
1.6.5 Computation of Position, Velocity and Time . . . 11
1.6.5.1 Dual-band RTK through NTRIP . . . 12
1.7 About the software license . . . 14
1.8 Publications and Credits . . . 14
1.9 Ok, now what? . . . 15

2 Reference Documents 17

2.1 Interface Control Documents . . . 17
2.1.1 GPS . . . 17
2.1.2 GLONASS . . . 17
2.1.3 Galileo . . . 18
2.1.4 BeiDou . . . 18
2.1.5 Satellite Based Augmentation Systems (SBAS) . . . 19
2.2 Other Standards . . . 19
2.2.1 RINEX . . . 19
2.2.2 NMEA . . . 20
2.2.3 KML . . . 20
2.2.4 C++ Standards . . . 20
2.2.5 Positioning protocols in wireless communication networks . . . 20

3 Signal model 23

3.1 GNSS signal model . . . 23
3.1.1 Global Positioning System (GPS) signal in space . . . 24
3.1.2 GLONASS signal in space . . . 25
3.1.3 Galileo signal in space . . . 26

3.1.4 Reference . . . 29

4 Todo List **31**

5 Topic Index **33**

5.1 Topics . . . 33

6 Namespace Index **35**

6.1 Namespace List . . . 35

7 Hierarchical Index **37**

7.1 Class Hierarchy . . . 37

8 Class Index **47**

8.1 Class List . . . 47

9 File Index **61**

9.1 File List . . . 61

10 Topic Documentation **79**

10.1 Acquisition . . . 79

10.1.1 Detailed Description . . . 79

10.1.2 acquisition_adapters . . . 79

10.1.2.1 Detailed Description . . . 79

10.1.3 acquisition_gr_blocks . . . 80

10.1.3.1 Detailed Description . . . 81

10.1.3.2 Typedef Documentation . . . 81

10.1.4 acquisition_libs . . . 82

10.1.4.1 Detailed Description . . . 82

10.2 Core GNSS Receiver . . . 83

10.2.1 Detailed Description . . . 83

10.2.2 GNSS block interfaces . . . 83

10.2.2.1 Detailed Description . . . 84

10.2.2.2 Typedef Documentation . . . 84

10.2.2.3 Function Documentation . . . 84

10.2.3 core_libs . . . 84

10.2.3.1 Detailed Description . . . 86

10.2.3.2 Macro Definition Documentation . . . 86

10.2.3.3 Typedef Documentation . . . 86

10.2.3.4 Function Documentation . . . 88

10.2.3.5 Variable Documentation . . . 88

10.2.4 core_monitor . . . 89

10.2.4.1 Detailed Description . . . 89

10.2.4.2 Typedef Documentation . . . 89

10.2.5 core_receiver . . . 89

10.2.5.1 Detailed Description . . . 90
10.2.5.2 Function Documentation . . . 90
10.2.6 core_system_parameters . . . 91
10.2.6.1 Detailed Description . . . 132
10.2.6.2 Macro Definition Documentation . . . 132
10.2.6.3 Typedef Documentation . . . 132
10.2.6.4 Enumeration Type Documentation . . . 133
10.2.6.5 Function Documentation . . . 133
10.2.6.6 Variable Documentation . . . 135
10.3 Channel . . . 227
10.3.1 Detailed Description . . . 227
10.3.2 channel_adapters . . . 227
10.3.2.1 Detailed Description . . . 227
10.3.3 channel_libs . . . 227
10.3.3.1 Detailed Description . . . 227
10.3.3.2 Typedef Documentation . . . 227
10.4 Signal Conditioner . . . 228
10.4.1 Detailed Description . . . 228
10.4.2 conditioner_adapters . . . 228
10.4.2.1 Detailed Description . . . 228
10.5 Data Type Adapters . . . 228
10.5.1 Detailed Description . . . 228
10.5.2 data_type_adapters . . . 228
10.5.2.1 Detailed Description . . . 229
10.5.3 data_type_gr_blocks . . . 229
10.5.3.1 Detailed Description . . . 229
10.5.3.2 Typedef Documentation . . . 229
10.6 Input Filter . . . 230
10.6.1 Detailed Description . . . 230
10.6.2 input_filter_adapters . . . 230
10.6.2.1 Detailed Description . . . 230
10.6.3 input_filter_gr_blocks . . . 230
10.6.3.1 Detailed Description . . . 231
10.6.3.2 Typedef Documentation . . . 231
10.6.3.3 Variable Documentation . . . 231
10.7 Algorithms Common Library . . . 231
10.7.1 Detailed Description . . . 231
10.7.2 algorithms_libs . . . 231
10.7.2.1 Detailed Description . . . 238
10.7.2.2 Typedef Documentation . . . 238
10.7.2.3 Function Documentation . . . 238
10.7.3 gnss_sdr_flags . . . 255

10.7.3.1 Detailed Description . . . 256
10.7.3.2 Function Documentation . . . 256
10.7.3.3 Variable Documentation . . . 259
10.8 PVT . . . 259
10.8.1 Detailed Description . . . 259
10.8.2 algorithms_libs_rtklib . . . 259
10.8.2.1 Detailed Description . . . 271
10.8.2.2 Macro Definition Documentation . . . 271
10.8.2.3 Typedef Documentation . . . 277
10.8.2.4 Function Documentation . . . 277
10.8.2.5 Variable Documentation . . . 278
10.8.3 RTKLIB . . . 297
10.8.3.1 Detailed Description . . . 297
10.8.4 pvt_adapters . . . 297
10.8.4.1 Detailed Description . . . 297
10.8.5 pvt_gr_blocks . . . 297
10.8.5.1 Detailed Description . . . 297
10.8.5.2 Typedef Documentation . . . 297
10.8.6 pvt_libs . . . 297
10.8.6.1 Detailed Description . . . 299
10.8.6.2 Typedef Documentation . . . 299
10.8.6.3 Enumeration Type Documentation . . . 300
10.8.6.4 Function Documentation . . . 300
10.9 Observables . . . 302
10.9.1 Detailed Description . . . 302
10.9.2 obs_adapters . . . 302
10.9.2.1 Detailed Description . . . 302
10.9.3 obs_gr_blocks . . . 302
10.9.3.1 Detailed Description . . . 303
10.9.3.2 Typedef Documentation . . . 303
10.9.4 observables_libs . . . 303
10.9.4.1 Detailed Description . . . 303
10.10 Resampler . . . 303
10.10.1 Detailed Description . . . 303
10.10.2 resampler_adapters . . . 303
10.10.2.1 Detailed Description . . . 303
10.10.3 resampler_gr_blocks . . . 303
10.10.3.1 Detailed Description . . . 304
10.10.3.2 Typedef Documentation . . . 304
10.11 Signal Source . . . 304
10.11.1 Detailed Description . . . 304
10.11.2 signal_source_adapters . . . 304

10.11.2.1 Detailed Description . . . 306
10.11.3 signal_source_gr_blocks . . . 306
10.11.3.1 Detailed Description . . . 307
10.11.3.2 Typedef Documentation . . . 307
10.11.4 signal_source_libs . . . 309
10.11.4.1 Detailed Description . . . 310
10.11.4.2 Macro Definition Documentation . . . 310
10.11.4.3 Enumeration Type Documentation . . . 310
10.11.4.4 Function Documentation . . . 310
10.12 Telemetry_Decoder . . . 311
10.12.1 Detailed Description . . . 311
10.12.2 Telemetry_Decoder_adapters . . . 311
10.12.2.1 Detailed Description . . . 311
10.12.3 telemetry_decoder_gr_blocks . . . 311
10.12.3.1 Detailed Description . . . 312
10.12.3.2 Typedef Documentation . . . 312
10.12.3.3 Enumeration Type Documentation . . . 313
10.12.4 telemetry_decoder_libswiftcnav . . . 313
10.12.4.1 Detailed Description . . . 313
10.12.4.2 Macro Definition Documentation . . . 314
10.12.5 telemetry_decoder_libs . . . 315
10.12.5.1 Detailed Description . . . 315
10.13 Tracking . . . 315
10.13.1 Detailed Description . . . 315
10.13.2 tracking_adapters . . . 315
10.13.2.1 Detailed Description . . . 315
10.13.3 tracking_gr_blocks . . . 316
10.13.3.1 Detailed Description . . . 316
10.13.3.2 Typedef Documentation . . . 316
10.13.4 tracking_libs . . . 317
10.13.4.1 Detailed Description . . . 318
10.13.4.2 Macro Definition Documentation . . . 319
10.13.4.3 Typedef Documentation . . . 319
10.13.4.4 Function Documentation . . . 319

11 Namespace Documentation **323**

11.1 block_factory Namespace Reference . . . 323
11.1.1 Detailed Description . . . 323
11.1.2 Function Documentation . . . 323
11.1.2.1 get_role_name() . . . 323
11.1.2.2 GetBlock() . . . 324
11.2 gnss_tow Namespace Reference . . . 324

11.2.1 Detailed Description . . . 324
11.2.2 Function Documentation . . . 324
11.2.2.1 add_ms() . . . 324
11.2.2.2 circular_error_ms() . . . 325
11.2.2.3 floor_div() . . . 325
11.2.2.4 floor_to_second_ms() . . . 325
11.2.2.5 sample_counter_delta() . . . 325
11.2.2.6 seconds_to_ms() . . . 325
11.2.2.7 to_ms() . . . 325
11.2.2.8 week_after_delta() . . . 325
11.2.2.9 wrap_ms() . . . 325
11.2.2.10 wrap_s() . . . 326
11.2.2.11 wrap_s_to_ms() . . . 326
11.2.3 Variable Documentation . . . 326
11.2.3.1 WEEK_MS . . . 326
11.2.3.2 WEEK_S . . . 326
11.3 std Namespace Reference . . . 326
11.3.1 Detailed Description . . . 330

12 Class Documentation **331**

12.1 sdr_gnss_packet_t::[struct].sats Struct Reference . . . 331
12.1.1 Detailed Description . . . 331
12.2 Acq_Conf Class Reference . . . 331
12.2.1 Detailed Description . . . 332
12.2.2 Member Function Documentation . . . 332
12.2.2.1 GetDwellResidualSamples() . . . 332
12.2.2.2 GetSamplesPerDwell() . . . 332
12.2.3 Member Data Documentation . . . 332
12.2.3.1 acquire_iq . . . 332
12.2.3.2 acquire_pilot . . . 332
12.2.3.3 bit_transition_flag . . . 333
12.2.3.4 blocking . . . 333
12.2.3.5 blocking_on_standby . . . 333
12.2.3.6 cboc . . . 333
12.2.3.7 chips_per_second . . . 333
12.2.3.8 code_length . . . 333
12.2.3.9 cuda_device . . . 333
12.2.3.10 doppler_max . . . 333
12.2.3.11 doppler_min . . . 333
12.2.3.12 doppler_step . . . 333
12.2.3.13 doppler_step2 . . . 333
12.2.3.14 dump . . . 333

12.2.3.15 dump_channel . . . 334
12.2.3.16 dump_filename . . . 334
12.2.3.17 enable_monitor_output . . . 334
12.2.3.18 folding_factor . . . 334
12.2.3.19 fs_in . . . 334
12.2.3.20 full_grid_search . . . 334
12.2.3.21 it_size . . . 334
12.2.3.22 item_type . . . 334
12.2.3.23 make_2_steps . . . 334
12.2.3.24 max_dwells . . . 334
12.2.3.25 ms_per_code . . . 334
12.2.3.26 num_codes . . . 334
12.2.3.27 num_doppler_bins_step2 . . . 335
12.2.3.28 pfa . . . 335
12.2.3.29 pfa2 . . . 335
12.2.3.30 qmboc . . . 335
12.2.3.31 reference_bin_min_sidelobes . . . 335
12.2.3.32 resampled_fs . . . 335
12.2.3.33 resampler_latency_samples . . . 335
12.2.3.34 resampler_ratio . . . 335
12.2.3.35 sampled_ms . . . 335
12.2.3.36 samples_per_chip . . . 335
12.2.3.37 samples_per_code . . . 335
12.2.3.38 samples_per_ms . . . 335
12.2.3.39 threshold . . . 336
12.2.3.40 use_automatic_resampler . . . 336
12.2.3.41 use_CFAR_algorithm_flag . . . 336
12.2.3.42 use_cuda . . . 336
12.2.3.43 vector_length . . . 336
12.2.3.44 zero_padding . . . 336
12.3 Acq_Conf_Fpga Class Reference . . . 336
12.3.1 Detailed Description . . . 337
12.3.2 Member Data Documentation . . . 337
12.3.2.1 all_fft_codes . . . 337
12.3.2.2 code_length . . . 337
12.3.2.3 code_length_chips . . . 337
12.3.2.4 code_rate_cps . . . 337
12.3.2.5 device_name . . . 337
12.3.2.6 doppler_max . . . 337
12.3.2.7 doppler_step . . . 337
12.3.2.8 doppler_step2 . . . 337
12.3.2.9 downsampling_factor . . . 337

12.3.2.10 downsampling_filter_delay . . . 338
12.3.2.11 downsampling_filter_num . . . 338
12.3.2.12 enable_zero_padding . . . 338
12.3.2.13 excludelimit . . . 338
12.3.2.14 fft_size . . . 338
12.3.2.15 fs_in . . . 338
12.3.2.16 make_2_steps . . . 338
12.3.2.17 max_num_acqs . . . 338
12.3.2.18 num_doppler_bins_step2 . . . 338
12.3.2.19 repeat_satellite . . . 338
12.3.2.20 resampled_fs . . . 338
12.3.2.21 threshold . . . 338
12.3.2.22 total_block_exp . . . 339
12.4 acquisition_impl_interface Class Reference . . . 339
12.4.1 Detailed Description . . . 339
12.4.2 Constructor & Destructor Documentation . . . 340
12.4.2.1 acquisition_impl_interface() . . . 340
12.4.3 Member Function Documentation . . . 340
12.4.3.1 mag() . . . 340
12.4.3.2 set_active() . . . 340
12.4.3.3 set_channel() . . . 340
12.4.3.4 set_channel_fsm() . . . 340
12.4.3.5 set_gnss_synchro() . . . 340
12.4.3.6 set_local_code() [1/2] . . . 340
12.4.3.7 set_local_code() [2/2] . . . 341
12.5 AcquisitionInterface Class Reference . . . 341
12.5.1 Detailed Description . . . 342
12.5.2 Member Function Documentation . . . 342
12.5.2.1 mag() . . . 342
12.5.2.2 reset() . . . 342
12.5.2.3 set_channel() . . . 342
12.5.2.4 set_channel_fsm() . . . 342
12.5.2.5 set_doppler_center() . . . 342
12.5.2.6 set_doppler_num_bins() . . . 342
12.5.2.7 set_gnss_synchro() . . . 343
12.5.2.8 set_local_code() . . . 343
12.5.2.9 set_resampler_latency() . . . 343
12.5.2.10 stop_acquisition() . . . 343
12.6 ad936x_iio_custom Class Reference . . . 343
12.6.1 Detailed Description . . . 343
12.6.2 Member Data Documentation . . . 344
12.6.2.1 n_channels . . . 344

12.7 ad936x_iio_samples Class Reference . . . 344
12.7.1 Detailed Description . . . 344
12.7.2 Member Data Documentation . . . 344
12.7.2.1 buffer . . . 344
12.7.2.2 n_bytes . . . 344
12.7.2.3 n_channels . . . 344
12.7.2.4 n_interleaved_iq_samples . . . 344
12.7.2.5 step_bytes . . . 344
12.8 ad936x_iio_source Class Reference . . . 344
12.8.1 Detailed Description . . . 345
12.8.2 Member Function Documentation . . . 345
12.8.2.1 start() . . . 345
12.8.2.2 stop() . . . 345
12.9 Ad936xCustomSignalSource Class Reference . . . 345
12.9.1 Detailed Description . . . 346
12.9.2 Member Function Documentation . . . 346
12.9.2.1 connect() . . . 346
12.9.2.2 disconnect() . . . 347
12.9.2.3 get_left_block() . . . 347
12.9.2.4 get_right_block() [1/2] . . . 347
12.9.2.5 get_right_block() [2/2] . . . 347
12.9.2.6 item_size() . . . 347
12.10 Adrv9361z7035SignalSourceFPGA Class Reference . . . 347
12.10.1 Detailed Description . . . 348
12.10.2 Member Function Documentation . . . 348
12.10.2.1 connect() . . . 348
12.10.2.2 disconnect() . . . 348
12.10.2.3 get_left_block() . . . 348
12.10.2.4 get_right_block() . . . 348
12.10.2.5 item_size() . . . 348
12.11 Agnss_Ref_Location Class Reference . . . 349
12.11.1 Detailed Description . . . 349
12.11.2 Constructor & Destructor Documentation . . . 349
12.11.2.1 Agnss_Ref_Location() . . . 349
12.11.3 Member Function Documentation . . . 349
12.11.3.1 serialize() . . . 349
12.11.4 Member Data Documentation . . . 349
12.11.4.1 lat . . . 349
12.11.4.2 lon . . . 349
12.11.4.3 uncertainty . . . 349
12.11.4.4 valid . . . 350
12.12 Agnss_Ref_Time Class Reference . . . 350

12.12.1 Detailed Description . . . 350
12.12.2 Constructor & Destructor Documentation . . . 350
12.12.2.1 Agnss_Ref_Time() . . . 350
12.12.3 Member Function Documentation . . . 350
12.12.3.1 serialize() . . . 350
12.12.4 Member Data Documentation . . . 350
12.12.4.1 microseconds . . . 350
12.12.4.2 seconds . . . 351
12.12.4.3 tow . . . 351
12.12.4.4 valid . . . 351
12.12.4.5 week . . . 351
12.13 Allocator Struct Reference . . . 351
12.13.1 Detailed Description . . . 351
12.13.2 Constructor & Destructor Documentation . . . 351
12.13.2.1 Allocator() . . . 351
12.13.3 Member Function Documentation . . . 351
12.13.3.1 operator()() . . . 351
12.13.4 Member Data Documentation . . . 352
12.13.4.1 buffer_ . . . 352
12.13.4.2 countwords_ . . . 352
12.14 alm_t Struct Reference . . . 352
12.14.1 Detailed Description . . . 352
12.14.2 Member Data Documentation . . . 352
12.14.2.1 A . . . 352
12.14.2.2 e . . . 352
12.14.2.3 f0 . . . 352
12.14.2.4 f1 . . . 352
12.14.2.5 i0 . . . 353
12.14.2.6 M0 . . . 353
12.14.2.7 omg . . . 353
12.14.2.8 OMG0 . . . 353
12.14.2.9 OMGd . . . 353
12.14.2.10 sat . . . 353
12.14.2.11 svconf . . . 353
12.14.2.12 svh . . . 353
12.14.2.13 toa . . . 353
12.14.2.14 toas . . . 353
12.14.2.15 week . . . 353
12.15 ambc_t Struct Reference . . . 354
12.15.1 Detailed Description . . . 354
12.15.2 Member Data Documentation . . . 354
12.15.2.1 epoch . . . 354

12.15.2.2 fixcnt . . . 354
12.15.2.3 flags . . . 354
12.15.2.4 LC . . . 354
12.15.2.5 LCv . . . 354
12.15.2.6 n . . . 354
12.16 An_Packet_Printer Class Reference . . . 354
12.16.1 Detailed Description . . . 355
12.16.2 Constructor & Destructor Documentation . . . 355
12.16.2.1 An_Packet_Printer() . . . 355
12.16.2.2 ∼An_Packet_Printer() . . . 355
12.16.3 Member Function Documentation . . . 355
12.16.3.1 close_serial() . . . 355
12.16.3.2 print_packet() . . . 355
12.17 an_packet_t Struct Reference . . . 355
12.17.1 Detailed Description . . . 355
12.17.2 Member Data Documentation . . . 356
12.17.2.1 data . . . 356
12.17.2.2 header . . . 356
12.18 ArraySignalConditioner Class Reference . . . 356
12.18.1 Detailed Description . . . 357
12.18.2 Constructor & Destructor Documentation . . . 357
12.18.2.1 ArraySignalConditioner() . . . 357
12.18.2.2 ∼ArraySignalConditioner() . . . 357
12.18.3 Member Function Documentation . . . 357
12.18.3.1 connect() . . . 357
12.18.3.2 data_type_adapter() . . . 357
12.18.3.3 disconnect() . . . 357
12.18.3.4 get_left_block() . . . 357
12.18.3.5 get_right_block() . . . 357
12.18.3.6 implementation() . . . 357
12.18.3.7 input_filter() . . . 358
12.18.3.8 item_size() . . . 358
12.18.3.9 resampler() . . . 358
12.18.3.10 role() . . . 358
12.19 Bayesian_estimator Class Reference . . . 358
12.19.1 Detailed Description . . . 358
12.20 Bds3_B1c_AlmanacMedium Struct Reference . . . 359
12.20.1 Detailed Description . . . 359
12.20.2 Member Data Documentation . . . 359
12.20.2.1 af0_s . . . 359
12.20.2.2 af1_s_s . . . 359
12.20.2.3 delta_i_rad . . . 359

12.20.2.4 eccentricity . . . 359
12.20.2.5 health . . . 359
12.20.2.6 m0_rad . . . 359
12.20.2.7 omega0_rad . . . 359
12.20.2.8 omega_dot_rad_s . . . 360
12.20.2.9 omega_rad . . . 360
12.20.2.10 prn . . . 360
12.20.2.11 sat_type . . . 360
12.20.2.12 sqrt_a_m_sqrt . . . 360
12.20.2.13 toa_s . . . 360
12.20.2.14 wna . . . 360
12.21 Bds3_B1c_AlmanacReduced Struct Reference . . . 360
12.21.1 Detailed Description . . . 360
12.21.2 Member Data Documentation . . . 360
12.21.2.1 delta_a_m . . . 360
12.21.2.2 health . . . 361
12.21.2.3 omega0_rad . . . 361
12.21.2.4 phi0_rad . . . 361
12.21.2.5 prn . . . 361
12.21.2.6 sat_type . . . 361
12.22 Bds3_B1c_Bgto Struct Reference . . . 361
12.22.1 Detailed Description . . . 361
12.22.2 Member Data Documentation . . . 361
12.22.2.1 a0_bgto_s . . . 361
12.22.2.2 a1_bgto_s_s . . . 361
12.22.2.3 a2_bgto_s_s2 . . . 361
12.22.2.4 gnss_id . . . 362
12.22.2.5 t0_bgto_s . . . 362
12.22.2.6 wn0_bgto . . . 362
12.23 Bds3_B1c_Eop Struct Reference . . . 362
12.23.1 Detailed Description . . . 362
12.23.2 Member Data Documentation . . . 362
12.23.2.1 delta_ut1_dot_s_day . . . 362
12.23.2.2 delta_ut1_s . . . 362
12.23.2.3 pm_x_arcsec . . . 362
12.23.2.4 pm_x_dot_arcsec_day . . . 362
12.23.2.5 pm_y_arcsec . . . 362
12.23.2.6 pm_y_dot_arcsec_day . . . 363
12.23.2.7 t_eop_s . . . 363
12.24 Bds3_B1c_PageCommon Struct Reference . . . 363
12.24.1 Detailed Description . . . 363
12.24.2 Member Data Documentation . . . 363

12.24.2.1 aif . . . 363
12.24.2.2 dif . . . 363
12.24.2.3 hs . . . 363
12.24.2.4 page_id . . . 363
12.24.2.5 sif . . . 363
12.24.2.6 sismai . . . 364
12.25 Bds3_B1c_PageData Struct Reference . . . 364
12.25.1 Detailed Description . . . 364
12.25.2 Member Data Documentation . . . 364
12.25.2.1 bgto . . . 364
12.25.2.2 common . . . 364
12.25.2.3 eop . . . 364
12.25.2.4 medium_almanac . . . 364
12.25.2.5 reduced_almanac . . . 364
12.25.2.6 sisaioc . . . 364
12.25.2.7 sisaioe . . . 365
12.25.2.8 toa_s . . . 365
12.25.2.9 wna . . . 365
12.26 beamformer Class Reference . . . 365
12.26.1 Detailed Description . . . 365
12.27 BeamformerFilter Class Reference . . . 365
12.27.1 Detailed Description . . . 366
12.27.2 Member Function Documentation . . . 366
12.27.2.1 connect() . . . 366
12.27.2.2 disconnect() . . . 366
12.27.2.3 get_left_block() . . . 366
12.27.2.4 get_right_block() . . . 366
12.27.2.5 implementation() . . . 367
12.27.2.6 item_size() . . . 367
12.27.2.7 role() . . . 367
12.28 beidou_b1c_telemetry_decoder_gs Class Reference . . . 367
12.28.1 Detailed Description . . . 368
12.28.2 Member Function Documentation . . . 368
12.28.2.1 reset() . . . 368
12.28.2.2 set_channel() . . . 368
12.28.2.3 set_satellite() . . . 368
12.29 beidou_b2a_telemetry_decoder_gs Class Reference . . . 368
12.29.1 Detailed Description . . . 369
12.29.2 Member Function Documentation . . . 369
12.29.2.1 reset() . . . 369
12.29.2.2 set_channel() . . . 369
12.29.2.3 set_satellite() . . . 369

12.30 Beidou_Cnav1_Ephemeris Class Reference . . . 369
12.30.1 Detailed Description . . . 372
12.30.2 Constructor & Destructor Documentation . . . 372
12.30.2.1 Beidou_Cnav1_Ephemeris() . . . 372
12.30.3 Member Function Documentation . . . 372
12.30.3.1 serialize() . . . 372
12.30.4 Member Data Documentation . . . 372
12.30.4.1 hs . . . 372
12.30.4.2 IODC . . . 372
12.30.4.3 IODE . . . 372
12.30.4.4 ISC_B1Cd . . . 372
12.30.4.5 ISC_B2ad . . . 372
12.30.4.6 nav_type . . . 373
12.30.4.7 sat_type . . . 373
12.30.4.8 sig_type . . . 373
12.30.4.9 TGD_B1Cp . . . 373
12.30.4.10 TGD_B2ap . . . 373
12.31 Beidou_Cnav1_Iono Class Reference . . . 373
12.31.1 Detailed Description . . . 373
12.31.2 Member Data Documentation . . . 373
12.31.2.1 alpha1 . . . 373
12.31.2.2 alpha2 . . . 374
12.31.2.3 alpha3 . . . 374
12.31.2.4 alpha4 . . . 374
12.31.2.5 alpha5 . . . 374
12.31.2.6 alpha6 . . . 374
12.31.2.7 alpha7 . . . 374
12.31.2.8 alpha8 . . . 374
12.31.2.9 alpha9 . . . 374
12.31.2.10 valid . . . 374
12.32 Beidou_Cnav1_Navigation_Message Class Reference . . . 374
12.32.1 Detailed Description . . . 375
12.32.2 Member Function Documentation . . . 375
12.32.2.1 get_last_nav_bits() . . . 375
12.33 Beidou_Cnav1_PageData_Message Struct Reference . . . 375
12.33.1 Detailed Description . . . 375
12.33.2 Member Data Documentation . . . 375
12.33.2.1 page_data . . . 375
12.33.2.2 PRN . . . 375
12.34 Beidou_Cnav1_Utc_Model Class Reference . . . 375
12.34.1 Detailed Description . . . 376
12.34.2 Member Data Documentation . . . 376

12.34.2.1 A0 . . . 376
12.34.2.2 A1 . . . 376
12.34.2.3 A2 . . . 376
12.34.2.4 delta_t_LS . . . 376
12.34.2.5 delta_t_LSF . . . 376
12.34.2.6 DN . . . 376
12.34.2.7 tot . . . 376
12.34.2.8 valid . . . 376
12.34.2.9 WN_LSF . . . 377
12.34.2.10 WN_t . . . 377
12.35 Beidou_Cnav2_Navigation_Message Class Reference . . . 377
12.35.1 Detailed Description . . . 377
12.35.2 Member Function Documentation . . . 377
12.35.2.1 clear_flags() . . . 377
12.35.2.2 get_ephemeris() . . . 377
12.35.2.3 get_last_nav_bits() . . . 377
12.35.2.4 have_new_ephemeris() . . . 377
12.35.2.5 last_crc_ok() . . . 378
12.35.2.6 last_frame_prn() . . . 378
12.35.2.7 last_mes_type() . . . 378
12.35.2.8 last_sow() . . . 378
12.35.2.9 preamble_detection_statistic() . . . 378
12.36 Beidou_Dnav_Almanac Class Reference . . . 378
12.36.1 Detailed Description . . . 380
12.36.2 Constructor & Destructor Documentation . . . 380
12.36.2.1 Beidou_Dnav_Almanac() . . . 380
12.36.3 Member Function Documentation . . . 380
12.36.3.1 serialize() . . . 380
12.36.4 Member Data Documentation . . . 380
12.36.4.1 AmEpID . . . 380
12.36.4.2 AmID . . . 380
12.36.4.3 expanded . . . 380
12.36.4.4 SV_health . . . 380
12.37 Beidou_Dnav_Differential_Correction Struct Reference . . . 380
12.37.1 Detailed Description . . . 381
12.37.2 Member Data Documentation . . . 381
12.37.2.1 delta_t . . . 381
12.37.2.2 delta_t_available . . . 381
12.37.2.3 PRN . . . 381
12.37.2.4 RURAI . . . 381
12.37.2.5 UDREI . . . 381
12.38 Beidou_Dnav_Differential_Corrections Class Reference . . . 381

12.38.1 Detailed Description . . . 382
12.38.2 Member Data Documentation . . . 382
12.38.2.1 BDEpID . . . 382
12.38.2.2 corrections . . . 382
12.38.2.3 expanded_corrections . . . 382
12.38.2.4 ionospheric_grid_valid . . . 382
12.38.2.5 Pnum2 . . . 382
12.38.2.6 range_corrections_valid . . . 382
12.38.2.7 SatH2 . . . 382
12.38.2.8 source_PRN . . . 382
12.38.2.9 SOW . . . 382
12.39 Beidou_Dnav_Ephemeris Class Reference . . . 383
12.39.1 Detailed Description . . . 385
12.39.2 Constructor & Destructor Documentation . . . 385
12.39.2.1 Beidou_Dnav_Ephemeris() . . . 385
12.39.3 Member Function Documentation . . . 386
12.39.3.1 serialize() . . . 386
12.39.4 Member Data Documentation . . . 386
12.39.4.1 alert_flag . . . 386
12.39.4.2 antispoofing_flag . . . 386
12.39.4.3 AODC . . . 386
12.39.4.4 AODE . . . 386
12.39.4.5 AODO . . . 386
12.39.4.6 fit_interval_flag . . . 386
12.39.4.7 integrity_status_flag . . . 387
12.39.4.8 nav_type . . . 387
12.39.4.9 satelliteBlock . . . 387
12.39.4.10 sig_type . . . 387
12.39.4.11 spare1 . . . 387
12.39.4.12 spare2 . . . 387
12.39.4.13 SV_accuracy . . . 387
12.39.4.14 SV_health . . . 387
12.39.4.15 TGD1 . . . 388
12.39.4.16 TGD2 . . . 388
12.40 Beidou_Dnav_Iono Class Reference . . . 388
12.40.1 Detailed Description . . . 389
12.40.2 Constructor & Destructor Documentation . . . 389
12.40.2.1 Beidou_Dnav_Iono() . . . 389
12.40.3 Member Function Documentation . . . 389
12.40.3.1 serialize() . . . 389
12.40.4 Member Data Documentation . . . 389
12.40.4.1 grid_points . . . 389

12.40.4.2 grid_valid . . . 389
12.41 Beidou_Dnav_Iono_Grid_Point Struct Reference . . . 390
12.41.1 Detailed Description . . . 390
12.41.2 Member Function Documentation . . . 390
12.41.2.1 serialize() . . . 390
12.41.3 Member Data Documentation . . . 390
12.41.3.1 available . . . 390
12.41.3.2 GIVEI . . . 390
12.41.3.3 monitored . . . 390
12.41.3.4 vertical_delay . . . 390
12.42 Beidou_Dnav_Navigation_Message Class Reference . . . 391
12.42.1 Detailed Description . . . 391
12.42.2 Constructor & Destructor Documentation . . . 391
12.42.2.1 Beidou_Dnav_Navigation_Message() . . . 391
12.42.3 Member Function Documentation . . . 392
12.42.3.1 d1_subframe_decoder() . . . 392
12.42.3.2 d2_subframe_decoder() . . . 392
12.42.3.3 get_almanac() . . . 392
12.42.3.4 get_differential_corrections() . . . 392
12.42.3.5 get_ephemeris() . . . 392
12.42.3.6 get_flag_CRC_test() . . . 392
12.42.3.7 get_flag_new_SOW_available() . . . 392
12.42.3.8 get_iono() . . . 392
12.42.3.9 get_SOW() . . . 392
12.42.3.10 get_utc_model() . . . 392
12.42.3.11 have_new_almanac() . . . 392
12.42.3.12 have_new_differential_corrections() . . . 393
12.42.3.13 have_new_ephemeris() . . . 393
12.42.3.14 have_new_iono() . . . 393
12.42.3.15 have_new_utc_model() . . . 393
12.42.3.16 set_flag_new_SOW_available() . . . 393
12.42.3.17 set_satellite_PRN() . . . 393
12.42.3.18 set_signal_type() . . . 393
12.42.3.19 utc_time() . . . 393
12.43 beidou_dnav_telemetry_decoder_gs Class Reference . . . 393
12.43.1 Detailed Description . . . 394
12.43.2 Constructor & Destructor Documentation . . . 394
12.43.2.1 ~beidou_dnav_telemetry_decoder_gs() . . . 394
12.43.3 Member Function Documentation . . . 395
12.43.3.1 general_work() . . . 395
12.43.3.2 reset() . . . 395
12.43.3.3 set_channel() . . . 395

12.43.3.4 set_satellite() . . . 395
12.44 Beidou_Dnav_Utc_Model Class Reference . . . 395
12.44.1 Detailed Description . . . 396
12.44.2 Member Function Documentation . . . 396
12.44.2.1 serialize() . . . 396
12.44.3 Member Data Documentation . . . 396
12.44.3.1 A0_GAL . . . 396
12.44.3.2 A0_GLO . . . 396
12.44.3.3 A0_GPS . . . 396
12.44.3.4 A0_UTC . . . 396
12.44.3.5 A1_GAL . . . 397
12.44.3.6 A1_GLO . . . 397
12.44.3.7 A1_GPS . . . 397
12.44.3.8 A1_UTC . . . 397
12.44.3.9 DeltaT_LS . . . 397
12.44.3.10 DeltaT_LSF . . . 397
12.44.3.11 DN . . . 397
12.44.3.12 valid . . . 397
12.44.3.13 WN_LSF . . . 397
12.45 BeidouLdpcGraph Struct Reference . . . 398
12.45.1 Detailed Description . . . 398
12.45.2 Member Data Documentation . . . 398
12.45.2.1 check_offsets . . . 398
12.45.2.2 check_to_h . . . 398
12.45.2.3 check_to_var . . . 398
12.45.2.4 num_checks . . . 398
12.45.2.5 num_variables . . . 398
12.45.2.6 row_weight . . . 398
12.45.2.7 var_offsets . . . 398
12.45.2.8 var_to_check . . . 399
12.45.2.9 var_to_edge . . . 399
12.45.2.10 var_to_h . . . 399
12.45.2.11 var_to_h_inv . . . 399
12.46 bladerf_source Class Reference . . . 399
12.46.1 Detailed Description . . . 399
12.47 BladerfSignalSource Class Reference . . . 400
12.47.1 Detailed Description . . . 400
12.47.2 Member Function Documentation . . . 401
12.47.2.1 connect() . . . 401
12.47.2.2 disconnect() . . . 401
12.47.2.3 get_left_block() . . . 401
12.47.2.4 get_right_block() . . . 401

12.47.2.5 item_size() . . . 401
12.48 byte_x2_to_complex_byte Class Reference . . . 401
12.48.1 Detailed Description . . . 401
12.49 ByteToShort Class Reference . . . 402
12.49.1 Detailed Description . . . 402
12.49.2 Member Function Documentation . . . 402
12.49.2.1 connect() . . . 402
12.49.2.2 disconnect() . . . 402
12.49.2.3 get_left_block() . . . 402
12.49.2.4 get_right_block() . . . 403
12.49.2.5 implementation() . . . 403
12.49.2.6 item_size() . . . 403
12.49.2.7 role() . . . 403
12.50 Channel Class Reference . . . 403
12.50.1 Detailed Description . . . 404
12.50.2 Constructor & Destructor Documentation . . . 404
12.50.2.1 Channel() . . . 404
12.50.2.2 ∼Channel() . . . 404
12.50.3 Member Function Documentation . . . 405
12.50.3.1 acquisition() . . . 405
12.50.3.2 assist_acquisition_doppler() . . . 405
12.50.3.3 connect() . . . 405
12.50.3.4 disconnect() . . . 405
12.50.3.5 get_left_block() . . . 405
12.50.3.6 get_left_block_acq() . . . 405
12.50.3.7 get_left_block_trk() . . . 405
12.50.3.8 get_right_block() . . . 405
12.50.3.9 get_right_block_acq() . . . 405
12.50.3.10 get_right_block_trk() . . . 405
12.50.3.11 get_signal() . . . 406
12.50.3.12 implementation() . . . 406
12.50.3.13 item_size() . . . 406
12.50.3.14 role() . . . 406
12.50.3.15 set_signal() . . . 406
12.50.3.16 start_acquisition() . . . 406
12.50.3.17 stop_channel() . . . 406
12.50.3.18 telemetry() . . . 406
12.50.3.19 tracking() . . . 406
12.51 Channel_Event Class Reference . . . 407
12.51.1 Detailed Description . . . 407
12.51.2 Member Data Documentation . . . 407
12.51.2.1 channel_id . . . 407

12.51.2.2 event_type . . . 407
12.52 channel_msg_receiver_cc Class Reference . . . 407
12.52.1 Detailed Description . . . 407
12.52.2 Constructor & Destructor Documentation . . . 408
12.52.2.1 ~channel_msg_receiver_cc() . . . 408
12.53 channel_status_msg_receiver Class Reference . . . 408
12.53.1 Detailed Description . . . 408
12.53.2 Constructor & Destructor Documentation . . . 408
12.53.2.1 ~channel_status_msg_receiver() . . . 408
12.53.3 Member Function Documentation . . . 409
12.53.3.1 clear_channel_status() . . . 409
12.53.3.2 get_current_status_map() . . . 409
12.53.3.3 get_current_status_pvt() . . . 409
12.53.3.4 get_current_tracking_map() . . . 409
12.54 ChannelFsm Class Reference . . . 409
12.54.1 Detailed Description . . . 409
12.55 ChannelInterface Class Reference . . . 410
12.55.1 Detailed Description . . . 410
12.55.2 Member Function Documentation . . . 410
12.55.2.1 get_left_block() . . . 410
12.55.2.2 get_left_block_acq() . . . 411
12.55.2.3 get_left_block_trk() . . . 411
12.55.2.4 get_right_block() . . . 411
12.55.2.5 get_right_block_acq() . . . 411
12.55.2.6 get_right_block_trk() . . . 411
12.55.2.7 set_signal() . . . 411
12.55.2.8 start_acquisition() . . . 411
12.55.2.9 stop_channel() . . . 411
12.56 cl_fft_plan Struct Reference . . . 411
12.56.1 Detailed Description . . . 412
12.56.2 Member Data Documentation . . . 412
12.56.2.1 context . . . 412
12.56.2.2 dim . . . 412
12.56.2.3 format . . . 412
12.56.2.4 kernel_info . . . 412
12.56.2.5 kernel_string . . . 412
12.56.2.6 last_batch_size . . . 412
12.56.2.7 max_localmem_fft_size . . . 412
12.56.2.8 max_radix . . . 412
12.56.2.9 max_work_item_per_workgroup . . . 412
12.56.2.10 min_mem_coalesce_width . . . 413
12.56.2.11 n . . . 413

12.56.2.12 num_kernels . . . 413
12.56.2.13 num_local_mem_banks . . . 413
12.56.2.14 program . . . 413
12.56.2.15 temp_buffer_needed . . . 413
12.56.2.16 tempmemobj . . . 413
12.56.2.17 tempmemobj_imag . . . 413
12.56.2.18 tempmemobj_real . . . 413
12.56.2.19 twist_kernel . . . 413
12.57 clFFT_Complex Struct Reference . . . 413
12.57.1 Detailed Description . . . 414
12.57.2 Member Data Documentation . . . 414
12.57.2.1 imag . . . 414
12.57.2.2 real . . . 414
12.58 clFFT_Dim3 Struct Reference . . . 414
12.58.1 Detailed Description . . . 414
12.58.2 Member Data Documentation . . . 414
12.58.2.1 x . . . 414
12.58.2.2 y . . . 414
12.58.2.3 z . . . 414
12.59 clFFT_SplitComplex Struct Reference . . . 414
12.59.1 Detailed Description . . . 415
12.59.2 Member Data Documentation . . . 415
12.59.2.1 imag . . . 415
12.59.2.2 real . . . 415
12.60 cnav_msg_decoder_t Struct Reference . . . 415
12.60.1 Detailed Description . . . 415
12.60.2 Member Data Documentation . . . 415
12.60.2.1 part1 . . . 415
12.60.2.2 part2 . . . 415
12.61 cnav_msg_t Struct Reference . . . 415
12.61.1 Detailed Description . . . 416
12.61.2 Member Data Documentation . . . 416
12.61.2.1 alert . . . 416
12.61.2.2 msg_id . . . 416
12.61.2.3 prn . . . 416
12.61.2.4 raw_msg . . . 416
12.61.2.5 tow . . . 416
12.62 cnav_v27_part_t Struct Reference . . . 416
12.62.1 Detailed Description . . . 417
12.62.2 Member Data Documentation . . . 417
12.62.2.1 crc_ok . . . 417
12.62.2.2 dec . . . 417

12.62.2.3 decisions . . . 417
12.62.2.4 decoded . . . 417
12.62.2.5 init . . . 417
12.62.2.6 invert . . . 417
12.62.2.7 message_lock . . . 418
12.62.2.8 n_crc_fail . . . 418
12.62.2.9 n_decoded . . . 418
12.62.2.10 n_symbols . . . 418
12.62.2.11 preamble_seen . . . 418
12.62.2.12 symbols . . . 418
12.63 Command_Event Class Reference . . . 418
12.63.1 Detailed Description . . . 418
12.63.2 Member Data Documentation . . . 418
12.63.2.1 command_id . . . 418
12.63.2.2 event_type . . . 419
12.64 complex_byte_to_float_x2 Class Reference . . . 419
12.64.1 Detailed Description . . . 419
12.65 complex_float_to_complex_byte Class Reference . . . 419
12.65.1 Detailed Description . . . 420
12.66 Concurrent_Map< Data > Class Template Reference . . . 420
12.66.1 Detailed Description . . . 420
12.66.2 Member Function Documentation . . . 420
12.66.2.1 get_map_copy() [1/2] . . . 420
12.66.2.2 get_map_copy() [2/2] . . . 420
12.66.2.3 read() . . . 420
12.66.2.4 size() . . . 420
12.66.2.5 write() . . . 421
12.67 Concurrent_Queue< Data > Class Template Reference . . . 421
12.67.1 Detailed Description . . . 421
12.67.2 Member Function Documentation . . . 421
12.67.2.1 clear() . . . 421
12.67.2.2 empty() . . . 421
12.67.2.3 push() [1/2] . . . 421
12.67.2.4 push() [2/2] . . . 422
12.67.2.5 size() . . . 422
12.67.2.6 timed_wait_and_pop() . . . 422
12.67.2.7 try_pop() . . . 422
12.67.2.8 wait_and_pop() . . . 422
12.68 HistogramBitSynchronizer::Config Struct Reference . . . 422
12.68.1 Detailed Description . . . 423
12.68.2 Constructor & Destructor Documentation . . . 423
12.68.2.1 Config() . . . 423

12.68.3 Member Data Documentation . . . 423
12.68.3.1 bit_period_ms . . . 423
12.68.3.2 dominance_ratio . . . 423
12.68.3.3 epoch_ms . . . 423
12.68.3.4 min_events_for_lock . . . 424
12.68.3.5 min_prompt_mag . . . 424
12.68.3.6 runner_up_margin . . . 424
12.68.3.7 stable_best_required . . . 424
12.68.3.8 tentative_events_required . . . 424
12.68.3.9 transition_confidence . . . 424
12.68.3.10 transition_window_epochs . . . 425
12.68.3.11 use_phase_dot_detector . . . 425
12.69 ConfigurationInterface Class Reference . . . 425
12.69.1 Detailed Description . . . 426
12.70 conjugate_cc Class Reference . . . 426
12.70.1 Detailed Description . . . 426
12.71 conjugate_ic Class Reference . . . 426
12.71.1 Detailed Description . . . 427
12.72 conjugate_sc Class Reference . . . 427
12.72.1 Detailed Description . . . 427
12.73 ControlThread Class Reference . . . 427
12.73.1 Detailed Description . . . 428
12.73.2 Constructor & Destructor Documentation . . . 428
12.73.2.1 ControlThread() [1/2] . . . 428
12.73.2.2 ControlThread() [2/2] . . . 428
12.73.2.3 ~ControlThread() . . . 428
12.73.3 Member Function Documentation . . . 428
12.73.3.1 applied_actions() . . . 428
12.73.3.2 flowgraph() . . . 428
12.73.3.3 processed_control_messages() . . . 429
12.73.3.4 run() . . . 429
12.73.3.5 set_control_queue() . . . 429
12.73.4 Member Data Documentation . . . 429
12.73.4.1 me . . . 429
12.74 Cpu_Multicorrelator Class Reference . . . 429
12.74.1 Detailed Description . . . 430
12.75 Cpu_Multicorrelator_16sc Class Reference . . . 430
12.75.1 Detailed Description . . . 430
12.76 Cpu_Multicorrelator_Real_Codes Class Reference . . . 430
12.76.1 Detailed Description . . . 431
12.77 cshort_to_float_x2 Class Reference . . . 431
12.77.1 Detailed Description . . . 431

12.78 cshort_to_gr_complex Class Reference . . . 431
12.78.1 Detailed Description . . . 432
12.79 CshortToGrComplex Class Reference . . . 432
12.79.1 Detailed Description . . . 432
12.79.2 Member Function Documentation . . . 432
12.79.2.1 connect() . . . 432
12.79.2.2 disconnect() . . . 433
12.79.2.3 get_left_block() . . . 433
12.79.2.4 get_right_block() . . . 433
12.79.2.5 implementation() . . . 433
12.79.2.6 item_size() . . . 433
12.79.2.7 role() . . . 433
12.80 CubatureFilter Class Reference . . . 433
12.80.1 Detailed Description . . . 433
12.81 cuda_multicorrelator Class Reference . . . 434
12.81.1 Detailed Description . . . 434
12.82 CudaPcpsEngine Class Reference . . . 434
12.82.1 Detailed Description . . . 435
12.82.2 Member Enumeration Documentation . . . 435
12.82.2.1 GridId . . . 435
12.82.3 Constructor & Destructor Documentation . . . 435
12.82.3.1 CudaPcpsEngine() . . . 435
12.82.4 Member Function Documentation . . . 435
12.82.4.1 compute_grid() . . . 435
12.82.4.2 device_bytes() . . . 436
12.82.4.3 device_name() . . . 436
12.82.4.4 is_valid() . . . 436
12.82.4.5 last_error() . . . 436
12.82.4.6 set_doppler_wipeoffs() . . . 436
12.82.4.7 set_fft_codes() . . . 437
12.83 CustomUDPSignalSource Class Reference . . . 437
12.83.1 Detailed Description . . . 438
12.83.2 Member Function Documentation . . . 438
12.83.2.1 connect() . . . 438
12.83.2.2 disconnect() . . . 438
12.83.2.3 get_left_block() . . . 438
12.83.2.4 get_right_block() [1/2] . . . 438
12.83.2.5 get_right_block() [2/2] . . . 438
12.83.2.6 item_size() . . . 438
12.84 Deleter Struct Reference . . . 438
12.84.1 Detailed Description . . . 439
12.84.2 Constructor & Destructor Documentation . . . 439

12.84.2.1 Deleter() . . . 439
12.84.3 Member Function Documentation . . . 439
12.84.3.1 operator()() . . . 439
12.84.4 Member Data Documentation . . . 439
12.84.4.1 buffer_ . . . 439
12.85 dgps_t Struct Reference . . . 439
12.85.1 Detailed Description . . . 439
12.85.2 Member Data Documentation . . . 439
12.85.2.1 iod . . . 439
12.85.2.2 prc . . . 439
12.85.2.3 rrc . . . 439
12.85.2.4 t0 . . . 440
12.85.2.5 udre . . . 440
12.86 direct_resampler_conditioner_cb Class Reference . . . 440
12.86.1 Detailed Description . . . 440
12.86.2 Member Function Documentation . . . 440
12.86.2.1 sample_freq_in() . . . 440
12.86.2.2 sample_freq_out() . . . 440
12.87 direct_resampler_conditioner_cc Class Reference . . . 441
12.87.1 Detailed Description . . . 441
12.87.2 Member Function Documentation . . . 441
12.87.2.1 sample_freq_in() . . . 441
12.87.2.2 sample_freq_out() . . . 441
12.88 direct_resampler_conditioner_cs Class Reference . . . 441
12.88.1 Detailed Description . . . 442
12.88.2 Member Function Documentation . . . 442
12.88.2.1 sample_freq_in() . . . 442
12.88.2.2 sample_freq_out() . . . 442
12.89 DirectResamplerConditioner Class Reference . . . 442
12.89.1 Detailed Description . . . 443
12.89.2 Member Function Documentation . . . 443
12.89.2.1 connect() . . . 443
12.89.2.2 disconnect() . . . 443
12.89.2.3 get_left_block() . . . 443
12.89.2.4 get_right_block() . . . 443
12.89.2.5 implementation() . . . 443
12.89.2.6 item_size() . . . 443
12.89.2.7 role() . . . 443
12.90 Dll_Pll_Conf Class Reference . . . 444
12.90.1 Detailed Description . . . 445
12.90.2 Member Data Documentation . . . 445
12.90.2.1 b1c_data_prompt_scale . . . 445

12.90.2.2 b1c_pilot_prompt_scale . . . 445
12.90.2.3 b1c_prompt_normalize_power . . . 445
12.90.2.4 b1c_prompt_use_data_q . . . 445
12.90.2.5 b1c_secondary_lock_ratio . . . 445
12.90.2.6 bit_synchronization_time_limit_s . . . 445
12.90.2.7 bs_dominance_ratio . . . 445
12.90.2.8 bs_min_events_for_lock . . . 446
12.90.2.9 bs_min_prompt_mag . . . 446
12.90.2.10 bs_runner_up_margin . . . 446
12.90.2.11 bs_stable_best_required . . . 446
12.90.2.12 bs_tentative_events_required . . . 446
12.90.2.13 bs_transition_confidence . . . 446
12.90.2.14 bs_transition_window_epochs . . . 446
12.90.2.15 bs_use_phase_dot_detector . . . 446
12.90.2.16 carrier_aiding . . . 446
12.90.2.17 carrier_lock_test_smoother_alpha . . . 446
12.90.2.18 carrier_lock_test_smoother_samples . . . 446
12.90.2.19 carrier_lock_th . . . 446
12.90.2.20 cn0_min . . . 447
12.90.2.21 cn0_samples . . . 447
12.90.2.22 cn0_smoother_alpha . . . 447
12.90.2.23 cn0_smoother_samples . . . 447
12.90.2.24 dll_bw_hz . . . 447
12.90.2.25 dll_bw_narrow_hz . . . 447
12.90.2.26 dll_filter_order . . . 447
12.90.2.27 dll_pull_in_bw_hz . . . 447
12.90.2.28 dump . . . 447
12.90.2.29 dump_filename . . . 447
12.90.2.30 dump_mat . . . 447
12.90.2.31 early_late_space_chips . . . 447
12.90.2.32 early_late_space_narrow_chips . . . 448
12.90.2.33 enable_doppler_correction . . . 448
12.90.2.34 enable_fll_pull_in . . . 448
12.90.2.35 enable_fll_steady_state . . . 448
12.90.2.36 extend_correlation_symbols . . . 448
12.90.2.37 f_error_accumulation . . . 448
12.90.2.38 f_error_doppler_step . . . 448
12.90.2.39 f_error_dump . . . 448
12.90.2.40 f_error_dump_filename . . . 448
12.90.2.41 f_error_step_num . . . 448
12.90.2.42 fll_bw_hz . . . 448
12.90.2.43 fll_filter_order . . . 448

12.90.2.44 fs_in . . . 449
12.90.2.45 high_dyn . . . 449
12.90.2.46 item_type . . . 449
12.90.2.47 max_carrier_lock_fail . . . 449
12.90.2.48 max_code_lock_fail . . . 449
12.90.2.49 pll_bw_hz . . . 449
12.90.2.50 pll_bw_narrow_hz . . . 449
12.90.2.51 pll_filter_order . . . 449
12.90.2.52 pll_pull_in_bw_hz . . . 449
12.90.2.53 pull_in_time_s . . . 449
12.90.2.54 qmboc . . . 449
12.90.2.55 signal . . . 449
12.90.2.56 slope . . . 450
12.90.2.57 smoother_length . . . 450
12.90.2.58 spc . . . 450
12.90.2.59 system . . . 450
12.90.2.60 tow_to_trk . . . 450
12.90.2.61 track_pilot . . . 450
12.90.2.62 vector_length . . . 450
12.90.2.63 very_early_late_space_chips . . . 450
12.90.2.64 very_early_late_space_narrow_chips . . . 450
12.90.2.65 y_intercept . . . 450
12.91 Dll_Pll_Conf_Fpga Class Reference . . . 450
12.91.1 Detailed Description . . . 452
12.91.2 Member Data Documentation . . . 452
12.91.2.1 bit_synchronization_time_limit_s . . . 452
12.91.2.2 bs_dominance_ratio . . . 452
12.91.2.3 bs_min_events_for_lock . . . 452
12.91.2.4 bs_min_prompt_mag . . . 452
12.91.2.5 bs_runner_up_margin . . . 452
12.91.2.6 bs_stable_best_required . . . 452
12.91.2.7 bs_tentative_events_required . . . 452
12.91.2.8 bs_transition_confidence . . . 452
12.91.2.9 bs_transition_window_epochs . . . 452
12.91.2.10 bs_use_phase_dot_detector . . . 453
12.91.2.11 ca_codes . . . 453
12.91.2.12 carrier_aiding . . . 453
12.91.2.13 carrier_lock_test_smoother_alpha . . . 453
12.91.2.14 carrier_lock_test_smoother_samples . . . 453
12.91.2.15 carrier_lock_th . . . 453
12.91.2.16 cn0_min . . . 453
12.91.2.17 cn0_samples . . . 453

12.91.2.18 cn0_smoother_alpha . . . 453
12.91.2.19 cn0_smoother_samples . . . 453
12.91.2.20 code_length_chips . . . 453
12.91.2.21 code_samples_per_chip . . . 453
12.91.2.22 data_codes . . . 454
12.91.2.23 device_name . . . 454
12.91.2.24 dll_bw_hz . . . 454
12.91.2.25 dll_bw_narrow_hz . . . 454
12.91.2.26 dll_filter_order . . . 454
12.91.2.27 dll_pull_in_bw_hz . . . 454
12.91.2.28 dump . . . 454
12.91.2.29 dump_filename . . . 454
12.91.2.30 dump_mat . . . 454
12.91.2.31 early_late_space_chips . . . 454
12.91.2.32 early_late_space_narrow_chips . . . 454
12.91.2.33 enable_doppler_correction . . . 454
12.91.2.34 enable_fll_pull_in . . . 455
12.91.2.35 enable_fll_steady_state . . . 455
12.91.2.36 extend_correlation_symbols . . . 455
12.91.2.37 extend_fpga_integration_periods . . . 455
12.91.2.38 extended_correlation_in_fpga . . . 455
12.91.2.39 fll_bw_hz . . . 455
12.91.2.40 fll_filter_order . . . 455
12.91.2.41 fpga_integration_period . . . 455
12.91.2.42 fs_in . . . 455
12.91.2.43 high_dyn . . . 455
12.91.2.44 max_carrier_lock_fail . . . 455
12.91.2.45 max_code_lock_fail . . . 455
12.91.2.46 pll_bw_hz . . . 456
12.91.2.47 pll_bw_narrow_hz . . . 456
12.91.2.48 pll_filter_order . . . 456
12.91.2.49 pll_pull_in_bw_hz . . . 456
12.91.2.50 pull_in_time_s . . . 456
12.91.2.51 signal . . . 456
12.91.2.52 slope . . . 456
12.91.2.53 smoother_length . . . 456
12.91.2.54 spc . . . 456
12.91.2.55 system . . . 456
12.91.2.56 track_pilot . . . 456
12.91.2.57 vector_length . . . 456
12.91.2.58 very_early_late_space_chips . . . 457
12.91.2.59 very_early_late_space_narrow_chips . . . 457

12.91.2.60 y_intercept . . . 457
12.92 dll_pll_veml_tracking Class Reference . . . 457
12.92.1 Detailed Description . . . 457
12.92.2 Member Function Documentation . . . 458
12.92.2.1 f_error_bin_multiplier() . . . 458
12.92.3 Friends And Related Symbol Documentation . . . 458
12.92.3.1 DllPllTrackingFrequencyErrorTest . . . 458
12.93 dll_pll_veml_tracking_fpga Class Reference . . . 458
12.93.1 Detailed Description . . . 459
12.93.2 Constructor & Destructor Documentation . . . 459
12.93.2.1 ~dll_pll_veml_tracking_fpga() . . . 459
12.93.3 Member Function Documentation . . . 459
12.93.3.1 general_work() . . . 459
12.93.3.2 reset() . . . 459
12.93.3.3 set_channel() . . . 459
12.93.3.4 set_gnss_synchro() . . . 459
12.93.3.5 start_tracking() . . . 459
12.93.3.6 stop_tracking() . . . 459
12.94 DllPllTrackingAdapter Class Reference . . . 460
12.94.1 Detailed Description . . . 461
12.94.2 Constructor & Destructor Documentation . . . 461
12.94.2.1 DllPllTrackingAdapter() . . . 461
12.94.2.2 ~DllPllTrackingAdapter() . . . 461
12.94.3 Member Function Documentation . . . 461
12.94.3.1 connect() . . . 461
12.94.3.2 disconnect() . . . 461
12.94.3.3 get_left_block() . . . 461
12.94.3.4 get_right_block() . . . 461
12.94.3.5 implementation() . . . 461
12.94.3.6 item_size() . . . 462
12.94.3.7 role() . . . 462
12.94.3.8 set_channel() . . . 462
12.94.3.9 set_gnss_synchro() . . . 462
12.94.3.10 start_tracking() . . . 462
12.94.3.11 stop_tracking() . . . 462
12.95 DllPllTrackingAdapterFpga Class Reference . . . 462
12.95.1 Detailed Description . . . 463
12.95.2 Member Function Documentation . . . 463
12.95.2.1 connect() . . . 463
12.95.2.2 disconnect() . . . 463
12.95.2.3 get_left_block() . . . 463
12.95.2.4 get_right_block() . . . 463

12.95.2.5 implementation() . . . 463
12.95.2.6 item_size() . . . 464
12.95.2.7 role() . . . 464
12.95.2.8 set_channel() . . . 464
12.95.2.9 set_gnss_synchro() . . . 464
12.95.2.10 start_tracking() . . . 464
12.95.2.11 stop_tracking() . . . 464
12.96 DMASignalSourceFPGA Class Reference . . . 464
12.96.1 Detailed Description . . . 465
12.96.2 Member Function Documentation . . . 465
12.96.2.1 connect() . . . 465
12.96.2.2 disconnect() . . . 465
12.96.2.3 get_left_block() . . . 465
12.96.2.4 get_right_block() . . . 465
12.96.2.5 item_size() . . . 466
12.96.2.6 start() . . . 466
12.97 DSM_dsm_header Class Reference . . . 466
12.97.1 Detailed Description . . . 466
12.97.2 Member Data Documentation . . . 466
12.97.2.1 dsm_block_id . . . 466
12.97.2.2 dsm_id . . . 466
12.98 DSM_KROOT_message Class Reference . . . 466
12.98.1 Detailed Description . . . 467
12.98.2 Member Data Documentation . . . 467
12.98.2.1 alpha . . . 467
12.98.2.2 cidkr . . . 467
12.98.2.3 ds . . . 467
12.98.2.4 hf . . . 467
12.98.2.5 kroot . . . 467
12.98.2.6 ks . . . 467
12.98.2.7 maclt . . . 467
12.98.2.8 mf . . . 467
12.98.2.9 nb_dk . . . 467
12.98.2.10 p_dk . . . 467
12.98.2.11 pkid . . . 468
12.98.2.12 reserved . . . 468
12.98.2.13 reserved1 . . . 468
12.98.2.14 towh_k . . . 468
12.98.2.15 ts . . . 468
12.98.2.16 verified . . . 468
12.98.2.17 wn_k . . . 468
12.99 DSM_nma_header Class Reference . . . 468

12.99.1 Detailed Description . . . 468
12.99.2 Member Data Documentation . . . 468
12.99.2.1 cid . . . 468
12.99.2.2 cpks . . . 469
12.99.2.3 nmas . . . 469
12.99.2.4 reserved . . . 469
12.100 DSM_PKR_message Class Reference . . . 469
12.100.1 Detailed Description . . . 469
12.100.2 Member Data Documentation . . . 469
12.100.2.1 itn . . . 469
12.100.2.2 mid . . . 469
12.100.2.3 nb_dp . . . 469
12.100.2.4 npk . . . 469
12.100.2.5 npkt . . . 469
12.100.2.6 npktid . . . 470
12.100.2.7 p_dp . . . 470
12.101 eph_t Struct Reference . . . 470
12.101.1 Detailed Description . . . 471
12.101.2 Member Data Documentation . . . 471
12.101.2.1 A . . . 471
12.101.2.2 Adot . . . 471
12.101.2.3 apply_has_corrections . . . 471
12.101.2.4 cic . . . 471
12.101.2.5 cis . . . 471
12.101.2.6 cnav_top . . . 471
12.101.2.7 cnav_ura_valid . . . 471
12.101.2.8 cnav_uraed . . . 471
12.101.2.9 cnav_uraned0 . . . 471
12.101.2.10 cnav_uraned1 . . . 472
12.101.2.11 cnav_uraned2 . . . 472
12.101.2.12 cnav_wnop . . . 472
12.101.2.13 code . . . 472
12.101.2.14 crc . . . 472
12.101.2.15 crs . . . 472
12.101.2.16 cuc . . . 472
12.101.2.17 cus . . . 472
12.101.2.18 deln . . . 472
12.101.2.19 e . . . 472
12.101.2.20 f0 . . . 472
12.101.2.21 f1 . . . 472
12.101.2.22 f2 . . . 473
12.101.2.23 fit . . . 473

12.101.2.24 flag . . . 473
12.101.2.25 has_clock_correction_m . . . 473
12.101.2.26 has_orbit_cross_track_correction_m . . . 473
12.101.2.27 has_orbit_in_track_correction_m . . . 473
12.101.2.28 has_orbit_radial_correction_m . . . 473
12.101.2.29 i0 . . . 473
12.101.2.30 idot . . . 473
12.101.2.31 iodc . . . 473
12.101.2.32 iode . . . 473
12.101.2.33 isc . . . 473
12.101.2.34 M0 . . . 474
12.101.2.35 ndot . . . 474
12.101.2.36 omg . . . 474
12.101.2.37 OMG0 . . . 474
12.101.2.38 OMGd . . . 474
12.101.2.39 sat . . . 474
12.101.2.40 sva . . . 474
12.101.2.41 svh . . . 474
12.101.2.42 tgd . . . 474
12.101.2.43 toc . . . 474
12.101.2.44 toe . . . 474
12.101.2.45 toes . . . 474
12.101.2.46 ttr . . . 475
12.101.2.47 week . . . 475
12.102 erp_t Struct Reference . . . 475
12.102.1 Detailed Description . . . 475
12.102.2 Member Data Documentation . . . 475
12.102.2.1 data . . . 475
12.102.2.2 n . . . 475
12.102.2.3 nmax . . . 475
12.103 erpd_t Struct Reference . . . 475
12.103.1 Detailed Description . . . 475
12.103.2 Member Data Documentation . . . 476
12.103.2.1 lod . . . 476
12.103.2.2 mjd . . . 476
12.103.2.3 ut1_utc . . . 476
12.103.2.4 xp . . . 476
12.103.2.5 xpr . . . 476
12.103.2.6 yp . . . 476
12.103.2.7 ypr . . . 476
12.104 Evk1029SignalSource Class Reference . . . 476
12.104.1 Detailed Description . . . 477

12.104.2 Member Function Documentation . . . 477
12.104.2.1 connect() . . . 477
12.104.2.2 disconnect() . . . 477
12.104.2.3 get_right_block() . . . 477
12.104.2.4 item_size() . . . 478
12.105 Evk1029Source Class Reference . . . 478
12.105.1 Detailed Description . . . 478
12.106 Exponential_Smoother Class Reference . . . 478
12.106.1 Detailed Description . . . 479
12.106.2 Constructor & Destructor Documentation . . . 479
12.106.2.1 Exponential_Smoother() [1/2] . . . 479
12.106.2.2 ~Exponential_Smoother() . . . 479
12.106.2.3 Exponential_Smoother() [2/2] . . . 479
12.106.3 Member Function Documentation . . . 479
12.106.3.1 operator=() . . . 479
12.106.3.2 set_alpha() . . . 479
12.106.3.3 set_samples_for_initialization() . . . 479
12.107 exterr_t Struct Reference . . . 480
12.107.1 Detailed Description . . . 480
12.107.2 Member Data Documentation . . . 480
12.107.2.1 cerr . . . 480
12.107.2.2 ena . . . 480
12.107.2.3 gloicb . . . 480
12.107.2.4 gpsglob . . . 480
12.107.2.5 perr . . . 480
12.108 fcbd_t Struct Reference . . . 480
12.108.1 Detailed Description . . . 480
12.108.2 Member Data Documentation . . . 481
12.108.2.1 bias . . . 481
12.108.2.2 std . . . 481
12.108.2.3 te . . . 481
12.108.2.4 ts . . . 481
12.109 FifoReader Class Reference . . . 481
12.109.1 Detailed Description . . . 481
12.109.2 Member Typedef Documentation . . . 482
12.109.2.1 sptr . . . 482
12.109.3 Member Function Documentation . . . 482
12.109.3.1 start() . . . 482
12.110 FifoSignalSource Class Reference . . . 482
12.110.1 Detailed Description . . . 483
12.110.2 Member Function Documentation . . . 483
12.110.2.1 connect() . . . 483

12.110.2.2 disconnect() . . . 483
12.110.2.3 get_left_block() . . . 483
12.110.2.4 get_right_block() . . . 483
12.110.2.5 item_size() . . . 483
12.111 file_t Struct Reference . . . 484
12.111.1 Detailed Description . . . 484
12.111.2 Member Data Documentation . . . 484
12.111.2.1 fp . . . 484
12.111.2.2 fp_tag . . . 484
12.111.2.3 fp_tag_tmp . . . 484
12.111.2.4 fp_tmp . . . 484
12.111.2.5 fpos . . . 484
12.111.2.6 lock . . . 484
12.111.2.7 mode . . . 485
12.111.2.8 offset . . . 485
12.111.2.9 openpath . . . 485
12.111.2.10 path . . . 485
12.111.2.11 repmode . . . 485
12.111.2.12 speed . . . 485
12.111.2.13 start . . . 485
12.111.2.14 swapintv . . . 485
12.111.2.15 tick . . . 485
12.111.2.16 tick_f . . . 485
12.111.2.17 time . . . 485
12.111.2.18 timetag . . . 485
12.111.2.19 wtime . . . 486
12.112 FileConfiguration Class Reference . . . 486
12.112.1 Detailed Description . . . 486
12.112.2 Member Function Documentation . . . 486
12.112.2.1 is_present() . . . 486
12.112.2.2 property() [1/10] . . . 487
12.112.2.3 property() [2/10] . . . 487
12.112.2.4 property() [3/10] . . . 487
12.112.2.5 property() [4/10] . . . 487
12.112.2.6 property() [5/10] . . . 487
12.112.2.7 property() [6/10] . . . 487
12.112.2.8 property() [7/10] . . . 487
12.112.2.9 property() [8/10] . . . 487
12.112.2.10 property() [9/10] . . . 488
12.112.2.11 property() [10/10] . . . 488
12.112.2.12 set_property() . . . 488
12.113 FileSignalSource Class Reference . . . 488

12.113.1 Detailed Description . . . 490
12.114 FileSourceBase Class Reference . . . 490
12.114.1 Detailed Description . . . 493
12.114.2 Constructor & Destructor Documentation . . . 493
12.114.2.1 FileSourceBase() . . . 493
12.114.3 Member Function Documentation . . . 493
12.114.3.1 computeSamplesInFile() . . . 493
12.114.3.2 connect() . . . 493
12.114.3.3 disconnect() . . . 493
12.114.3.4 filename() . . . 493
12.114.3.5 get_left_block() . . . 494
12.114.3.6 get_right_block() . . . 494
12.114.3.7 init() . . . 494
12.114.3.8 item_size() . . . 494
12.114.3.9 item_type() . . . 494
12.114.3.10 itemTypeToSize() . . . 494
12.114.3.11 packetsPerSample() . . . 494
12.114.3.12 repeat() . . . 494
12.114.3.13 samples() . . . 494
12.114.3.14 samplesToSkip() . . . 494
12.114.3.15 sampling_frequency() . . . 494
12.114.3.16 source() . . . 495
12.114.3.17 source_item_size() . . . 495
12.115 FileTimestampSignalSource Class Reference . . . 495
12.115.1 Detailed Description . . . 497
12.115.2 Member Function Documentation . . . 497
12.115.2.1 create_file_source_hook() . . . 497
12.115.2.2 pre_connect_hook() . . . 497
12.115.2.3 pre_disconnect_hook() . . . 497
12.115.2.4 source() . . . 497
12.116 FirFilter Class Reference . . . 497
12.116.1 Detailed Description . . . 498
12.116.2 Constructor & Destructor Documentation . . . 498
12.116.2.1 FirFilter() . . . 498
12.116.2.2 ~FirFilter() . . . 498
12.116.3 Member Function Documentation . . . 499
12.116.3.1 connect() . . . 499
12.116.3.2 disconnect() . . . 499
12.116.3.3 get_left_block() . . . 499
12.116.3.4 get_right_block() . . . 499
12.116.3.5 implementation() . . . 499
12.116.3.6 item_size() . . . 499

12.116.3.7 role() . . . 499
12.117 FlexibandSignalSource Class Reference . . . 499
12.117.1 Detailed Description . . . 500
12.117.2 Member Function Documentation . . . 500
12.117.2.1 connect() . . . 500
12.117.2.2 disconnect() . . . 501
12.117.2.3 get_left_block() . . . 501
12.117.2.4 get_right_block() [1/2] . . . 501
12.117.2.5 get_right_block() [2/2] . . . 501
12.117.2.6 item_size() . . . 501
12.118 Fmcomms2SignalSource Class Reference . . . 501
12.118.1 Detailed Description . . . 502
12.118.2 Member Function Documentation . . . 502
12.118.2.1 connect() . . . 502
12.118.2.2 disconnect() . . . 502
12.118.2.3 get_left_block() . . . 502
12.118.2.4 get_right_block() . . . 502
12.118.2.5 item_size() . . . 502
12.119 Fmcomms5SignalSourceFPGA Class Reference . . . 503
12.119.1 Detailed Description . . . 503
12.119.2 Member Function Documentation . . . 504
12.119.2.1 connect() . . . 504
12.119.2.2 disconnect() . . . 504
12.119.2.3 get_left_block() . . . 504
12.119.2.4 get_right_block() . . . 504
12.119.2.5 item_size() . . . 504
12.120 FourBitCpxFileSignalSource Class Reference . . . 504
12.120.1 Detailed Description . . . 506
12.120.2 Member Function Documentation . . . 506
12.120.2.1 create_file_source_hook() . . . 506
12.120.2.2 itemTypeToSize() . . . 506
12.120.2.3 packetsPerSample() . . . 506
12.120.2.4 pre_connect_hook() . . . 507
12.120.2.5 pre_disconnect_hook() . . . 507
12.120.2.6 source() . . . 507
12.121 Fpga_Acquisition Class Reference . . . 507
12.121.1 Detailed Description . . . 508
12.121.2 Constructor & Destructor Documentation . . . 508
12.121.2.1 Fpga_Acquisition() . . . 508
12.121.2.2 ~Fpga_Acquisition() . . . 508
12.121.3 Member Function Documentation . . . 508
12.121.3.1 close_device() . . . 508

12.121.3.2 configure_acquisition() . . . 508
12.121.3.3 init() . . . 508
12.121.3.4 open_device() . . . 508
12.121.3.5 read_acquisition_results() . . . 509
12.121.3.6 reset_acquisition() . . . 509
12.121.3.7 run_acquisition() . . . 509
12.121.3.8 set_block_exp() . . . 509
12.121.3.9 set_doppler_sweep() . . . 509
12.121.3.10 set_local_code() . . . 509
12.121.3.11 stop_acquisition() . . . 509
12.121.3.12 write_local_code() . . . 509
12.122 Fpga_buffer_monitor Class Reference . . . 509
12.122.1 Detailed Description . . . 510
12.122.2 Constructor & Destructor Documentation . . . 510
12.122.2.1 Fpga_buffer_monitor() . . . 510
12.122.2.2 ~Fpga_buffer_monitor() . . . 510
12.122.3 Member Function Documentation . . . 510
12.122.3.1 check_buffer_overflow_and_monitor_buffer_status() . . . 510
12.123 Fpga_DMA Class Reference . . . 510
12.123.1 Detailed Description . . . 511
12.123.2 Constructor & Destructor Documentation . . . 511
12.123.2.1 Fpga_DMA() . . . 511
12.123.2.2 ~Fpga_DMA() . . . 511
12.123.3 Member Function Documentation . . . 511
12.123.3.1 DMA_close() . . . 511
12.123.3.2 DMA_open() . . . 511
12.123.3.3 DMA_write() . . . 511
12.123.3.4 get_buffer_address() . . . 511
12.123.3.5 get_buffer_size() . . . 511
12.124 Fpga_dynamic_bit_selection Class Reference . . . 512
12.124.1 Detailed Description . . . 512
12.124.2 Constructor & Destructor Documentation . . . 512
12.124.2.1 Fpga_dynamic_bit_selection() . . . 512
12.124.2.2 ~Fpga_dynamic_bit_selection() . . . 512
12.124.3 Member Function Documentation . . . 512
12.124.3.1 bit_selection() . . . 512
12.125 Fpga_Multicorrelator_8sc Class Reference . . . 512
12.125.1 Detailed Description . . . 513
12.125.2 Constructor & Destructor Documentation . . . 514
12.125.2.1 Fpga_Multicorrelator_8sc() . . . 514
12.125.2.2 ~Fpga_Multicorrelator_8sc() . . . 514
12.125.3 Member Function Documentation . . . 514

12.125.3.1 Carrier_wipeoff_multicorrelator_resampler() . . . 514
12.125.3.2 disable_secondary_codes() . . . 514
12.125.3.3 enable_secondary_codes() . . . 514
12.125.3.4 free() . . . 514
12.125.3.5 initialize_secondary_code() . . . 514
12.125.3.6 lock_channel() . . . 514
12.125.3.7 open_channel() . . . 515
12.125.3.8 read_sample_counter() . . . 515
12.125.3.9 set_initial_sample() . . . 515
12.125.3.10 set_local_code_and_taps() . . . 515
12.125.3.11 set_output_vectors() . . . 515
12.125.3.12 set_secondary_code_lengths() . . . 515
12.125.3.13 unlock_channel() . . . 515
12.125.3.14 update_local_code() . . . 515
12.125.3.15 update_prn_code_length() . . . 515
12.126 Fpga_spidev Class Reference . . . 516
12.126.1 Detailed Description . . . 516
12.126.2 Constructor & Destructor Documentation . . . 516
12.126.2.1 Fpga_spidev() . . . 516
12.126.2.2 ~Fpga_spidev() . . . 516
12.126.3 Member Function Documentation . . . 516
12.126.3.1 read_reg32() . . . 516
12.126.3.2 SPI_close() . . . 516
12.126.3.3 SPI_open() . . . 517
12.126.3.4 write_reg32() . . . 517
12.127 Fpga_Switch Class Reference . . . 517
12.127.1 Detailed Description . . . 517
12.127.2 Constructor & Destructor Documentation . . . 517
12.127.2.1 Fpga_Switch() . . . 517
12.127.2.2 ~Fpga_Switch() . . . 517
12.127.3 Member Function Documentation . . . 517
12.127.3.1 set_switch_position() . . . 517
12.128 FreqXlatingFirFilter Class Reference . . . 518
12.128.1 Detailed Description . . . 518
12.128.2 Member Function Documentation . . . 518
12.128.2.1 connect() . . . 518
12.128.2.2 disconnect() . . . 518
12.128.2.3 get_left_block() . . . 519
12.128.2.4 get_right_block() . . . 519
12.128.2.5 implementation() . . . 519
12.128.2.6 item_size() . . . 519
12.128.2.7 role() . . . 519

12.129 ftp_t Struct Reference . . . 519
12.129.1 Detailed Description . . . 519
12.129.2 Member Data Documentation . . . 519
12.129.2.1 addr . . . 519
12.129.2.2 error . . . 520
12.129.2.3 file . . . 520
12.129.2.4 local . . . 520
12.129.2.5 passwd . . . 520
12.129.2.6 proto . . . 520
12.129.2.7 state . . . 520
12.129.2.8 thread . . . 520
12.129.2.9 tnext . . . 520
12.129.2.10 topts . . . 520
12.129.2.11 user . . . 520
12.130 Galileo_Almanac Class Reference . . . 520
12.130.1 Detailed Description . . . 522
12.130.2 Constructor & Destructor Documentation . . . 522
12.130.2.1 Galileo_Almanac() . . . 522
12.130.3 Member Function Documentation . . . 522
12.130.3.1 serialize() . . . 522
12.130.4 Member Data Documentation . . . 522
12.130.4.1 E1B_HS . . . 522
12.130.4.2 E5a_HS . . . 522
12.130.4.3 E5b_HS . . . 522
12.130.4.4 IODa . . . 522
12.131 Galileo_Almanac_Helper Class Reference . . . 523
12.131.1 Detailed Description . . . 524
12.131.2 Constructor & Destructor Documentation . . . 524
12.131.2.1 Galileo_Almanac_Helper() . . . 524
12.131.3 Member Data Documentation . . . 524
12.131.3.1 af0_10 . . . 524
12.131.3.2 af0_8 . . . 524
12.131.3.3 af0_9 . . . 524
12.131.3.4 af1_10 . . . 524
12.131.3.5 af1_8 . . . 524
12.131.3.6 af1_9 . . . 524
12.131.3.7 DELTA_A_7 . . . 524
12.131.3.8 DELTA_A_8 . . . 525
12.131.3.9 DELTA_A_9 . . . 525
12.131.3.10 delta_i_7 . . . 525
12.131.3.11 delta_i_8 . . . 525
12.131.3.12 delta_i_9 . . . 525

12.131.3.13 E1B_HS_10 . . . 525
12.131.3.14 E1B_HS_8 . . . 525
12.131.3.15 E1B_HS_9 . . . 525
12.131.3.16 E5a_HS_10 . . . 525
12.131.3.17 E5a_HS_8 . . . 525
12.131.3.18 E5a_HS_9 . . . 525
12.131.3.19 E5b_HS_10 . . . 525
12.131.3.20 E5b_HS_8 . . . 526
12.131.3.21 E5b_HS_9 . . . 526
12.131.3.22 e_7 . . . 526
12.131.3.23 e_8 . . . 526
12.131.3.24 e_9 . . . 526
12.131.3.25 IOD_a_10 . . . 526
12.131.3.26 IOD_a_7 . . . 526
12.131.3.27 IOD_a_8 . . . 526
12.131.3.28 IOD_a_9 . . . 526
12.131.3.29 M0_10 . . . 526
12.131.3.30 M0_7 . . . 526
12.131.3.31 M0_9 . . . 526
12.131.3.32 Omega0_10 . . . 527
12.131.3.33 Omega0_7 . . . 527
12.131.3.34 Omega0_8 . . . 527
12.131.3.35 omega_7 . . . 527
12.131.3.36 omega_8 . . . 527
12.131.3.37 omega_9 . . . 527
12.131.3.38 Omega_dot_10 . . . 527
12.131.3.39 Omega_dot_7 . . . 527
12.131.3.40 Omega_dot_8 . . . 527
12.131.3.41 SVID1_7 . . . 527
12.131.3.42 SVID2_8 . . . 527
12.131.3.43 SVID3_9 . . . 527
12.131.3.44 t0a_7 . . . 528
12.131.3.45 t0a_9 . . . 528
12.131.3.46 WN_a_7 . . . 528
12.131.3.47 WN_a_9 . . . 528
12.132 Galileo_Cnav_Message Class Reference . . . 528
12.132.1 Detailed Description . . . 528
12.132.2 Member Function Documentation . . . 528
12.132.2.1 get_flag_CRC_test() . . . 528
12.132.2.2 get_HAS_encoded_page() . . . 528
12.132.2.3 have_new_HAS_page() . . . 529
12.132.2.4 is_HAS_in_test_mode() . . . 529

12.132.2.5 is_HAS_page_dummy() . . . 529
12.132.2.6 set_time_stamp() . . . 529
12.132.2.7 set_tow() . . . 529
12.132.2.8 set_week() . . . 529
12.133 Galileo_E1_Tcp_Connector_Tracking_cc Class Reference . . . 529
12.133.1 Detailed Description . . . 530
12.134 galileo_e5a_noncoherentIQ_acquisition_caf_cc Class Reference . . . 530
12.134.1 Detailed Description . . . 531
12.134.2 Constructor & Destructor Documentation . . . 531
12.134.2.1 ~galileo_e5a_noncoherentIQ_acquisition_caf_cc() . . . 531
12.134.3 Member Function Documentation . . . 531
12.134.3.1 general_work() . . . 531
12.134.3.2 mag() . . . 531
12.134.3.3 set_active() . . . 531
12.134.3.4 set_channel() . . . 531
12.134.3.5 set_channel_fsm() . . . 532
12.134.3.6 set_gnss_synchro() . . . 532
12.134.3.7 set_local_code() . . . 532
12.135 galileo_e6_has_msg_receiver Class Reference . . . 532
12.135.1 Detailed Description . . . 533
12.135.2 Constructor & Destructor Documentation . . . 533
12.135.2.1 ~galileo_e6_has_msg_receiver() . . . 533
12.135.3 Member Function Documentation . . . 533
12.135.3.1 process_test_page() . . . 533
12.136 Galileo_Ephemeris Class Reference . . . 533
12.136.1 Detailed Description . . . 536
12.136.2 Constructor & Destructor Documentation . . . 536
12.136.2.1 Galileo_Ephemeris() . . . 536
12.136.3 Member Function Documentation . . . 536
12.136.3.1 Galileo_System_Time() . . . 536
12.136.3.2 serialize() . . . 536
12.136.4 Member Data Documentation . . . 536
12.136.4.1 BGD_E1E5a . . . 536
12.136.4.2 BGD_E1E5b . . . 537
12.136.4.3 E1B_DVS . . . 537
12.136.4.4 E1B_HS . . . 537
12.136.4.5 E5a_DVS . . . 537
12.136.4.6 E5a_HS . . . 537
12.136.4.7 E5b_DVS . . . 537
12.136.4.8 E5b_HS . . . 537
12.136.4.9 flag_all_ephemeris . . . 537
12.136.4.10 IOD_ephemeris . . . 538

12.136.4.11 IOD_nav . . . 538
12.136.4.12 nav_message_source . . . 538
12.136.4.13 nav_message_type . . . 538
12.136.4.14 SISA . . . 538
12.137 Galileo_Ephemeris_Store Class Reference . . . 538
12.137.1 Detailed Description . . . 538
12.137.2 Member Typedef Documentation . . . 539
12.137.2.1 Ephemeris_Map . . . 539
12.137.3 Member Function Documentation . . . 539
12.137.3.1 by_source() . . . 539
12.137.3.2 clear() . . . 539
12.137.3.3 combined_view() . . . 539
12.137.3.4 empty() . . . 539
12.137.3.5 find() . . . 539
12.137.3.6 fnav() [1/2] . . . 539
12.137.3.7 fnav() [2/2] . . . 539
12.137.3.8 inav() [1/2] . . . 539
12.137.3.9 inav() [2/2] . . . 539
12.137.3.10 insert() . . . 540
12.138 Galileo_Fnav_Message Class Reference . . . 540
12.138.1 Detailed Description . . . 540
12.138.2 Member Function Documentation . . . 540
12.138.2.1 get_flag_CRC_test() . . . 540
12.138.2.2 get_flag_TOW_set() . . . 541
12.138.2.3 get_TOW1() . . . 541
12.138.2.4 get_TOW2() . . . 541
12.138.2.5 get_TOW3() . . . 541
12.138.2.6 get_TOW4() . . . 541
12.138.2.7 get_WN1() . . . 541
12.138.2.8 get_WN2() . . . 541
12.138.2.9 get_WN3() . . . 541
12.138.2.10 is_TOW1_set() . . . 541
12.138.2.11 is_TOW2_set() . . . 541
12.138.2.12 is_TOW3_set() . . . 541
12.138.2.13 is_TOW4_set() . . . 541
12.138.2.14 set_flag_TOW_set() . . . 542
12.138.2.15 set_TOW1_flag() . . . 542
12.138.2.16 set_TOW2_flag() . . . 542
12.138.2.17 set_TOW3_flag() . . . 542
12.138.2.18 set_TOW4_flag() . . . 542
12.139 Galileo_HAS_data Class Reference . . . 542
12.139.1 Detailed Description . . . 545

12.139.2 Member Function Documentation . . . 545
12.139.2.1 get_clock_correction_mult_m() . . . 545
12.139.2.2 get_clock_subset_correction_mult_m() . . . 546
12.139.2.3 get_code_bias_m() [1/2] . . . 546
12.139.2.4 get_code_bias_m() [2/2] . . . 546
12.139.2.5 get_delta_clock_correction_m() [1/2] . . . 546
12.139.2.6 get_delta_clock_correction_m() [2/2] . . . 546
12.139.2.7 get_delta_clock_subset_correction_m() [1/2] . . . 546
12.139.2.8 get_delta_clock_subset_correction_m() [2/2] . . . 546
12.139.2.9 get_delta_cross_track_m() [1/3] . . . 546
12.139.2.10 get_delta_cross_track_m() [2/3] . . . 546
12.139.2.11 get_delta_cross_track_m() [3/3] . . . 546
12.139.2.12 get_delta_in_track_m() [1/3] . . . 547
12.139.2.13 get_delta_in_track_m() [2/3] . . . 547
12.139.2.14 get_delta_in_track_m() [3/3] . . . 547
12.139.2.15 get_delta_radial_m() [1/3] . . . 547
12.139.2.16 get_delta_radial_m() [2/3] . . . 547
12.139.2.17 get_delta_radial_m() [3/3] . . . 547
12.139.2.18 get_gnss_id() . . . 547
12.139.2.19 get_gnss_iod() [1/2] . . . 547
12.139.2.20 get_gnss_iod() [2/2] . . . 547
12.139.2.21 get_nsat() . . . 547
12.139.2.22 get_nsat_sub() . . . 548
12.139.2.23 get_num_satellites() . . . 548
12.139.2.24 get_num_subset_satellites() . . . 548
12.139.2.25 get_phase_bias_cycle() [1/2] . . . 548
12.139.2.26 get_phase_bias_cycle() [2/2] . . . 548
12.139.2.27 get_phase_discontinuity_indicator() . . . 548
12.139.2.28 get_PRNs_in_mask() [1/2] . . . 548
12.139.2.29 get_PRNs_in_mask() [2/2] . . . 548
12.139.2.30 get_PRNs_in_submask() . . . 548
12.139.2.31 get_signals_in_mask() [1/2] . . . 548
12.139.2.32 get_signals_in_mask() [2/2] . . . 549
12.139.2.33 get_systems_string() . . . 549
12.139.2.34 get_systems_subset_string() . . . 549
12.139.2.35 get_time_of_message_s() . . . 549
12.139.2.36 get_validity_interval_s() . . . 549
12.139.2.37 has_validity_interval_at_tow() . . . 549
12.139.2.38 is_clock_do_not_use() . . . 549
12.139.2.39 is_clock_unavailable() . . . 549
12.139.2.40 is_code_bias_unavailable() . . . 549
12.139.2.41 is_orbit_along_cross_unavailable() . . . 549

12.139.2.42 is_orbit_radial_unavailable() . . . 550
12.139.2.43 is_phase_bias_unavailable() . . . 550
12.139.3 Member Data Documentation . . . 550
12.139.3.1 cell_mask . . . 550
12.139.3.2 cell_mask_availability_flag . . . 550
12.139.3.3 code_bias . . . 550
12.139.3.4 delta_clock_correction . . . 550
12.139.3.5 delta_clock_correction_clock_subset . . . 550
12.139.3.6 delta_clock_multiplier . . . 550
12.139.3.7 delta_clock_multiplier_clock_subset . . . 550
12.139.3.8 delta_cross_track . . . 550
12.139.3.9 delta_in_track . . . 551
12.139.3.10 delta_radial . . . 551
12.139.3.11 gnss_id_clock_subset . . . 551
12.139.3.12 gnss_id_mask . . . 551
12.139.3.13 gnss_iod . . . 551
12.139.3.14 has_status . . . 551
12.139.3.15 header . . . 551
12.139.3.16 message_id . . . 551
12.139.3.17 nav_message . . . 551
12.139.3.18 Nsys . . . 551
12.139.3.19 Nsys_sub . . . 552
12.139.3.20 phase_bias . . . 552
12.139.3.21 phase_discontinuity_indicator . . . 552
12.139.3.22 satellite_mask . . . 552
12.139.3.23 satellite_submask . . . 552
12.139.3.24 signal_mask . . . 552
12.139.3.25 tow . . . 552
12.139.3.26 validity_interval_index_clock_fullset_corrections . . . 552
12.139.3.27 validity_interval_index_clock_subset_corrections . . . 552
12.139.3.28 validity_interval_index_code_bias_corrections . . . 552
12.139.3.29 validity_interval_index_orbit_corrections . . . 553
12.139.3.30 validity_interval_index_phase_bias_corrections . . . 553
12.139.3.31 week . . . 553
12.140 Galileo_HAS_page Class Reference . . . 553
12.140.1 Detailed Description . . . 553
12.140.2 Member Data Documentation . . . 554
12.140.2.1 has_message_string . . . 554
12.140.2.2 has_status . . . 554
12.140.2.3 message_id . . . 554
12.140.2.4 message_page_id . . . 554
12.140.2.5 message_size . . . 554

12.140.2.6 message_type . . . 554
12.140.2.7 reserved . . . 554
12.140.2.8 time_stamp . . . 554
12.140.2.9 tow . . . 554
12.140.2.10 week . . . 555
12.141 Galileo_Inav_Message Class Reference . . . 555
12.141.1 Detailed Description . . . 556
12.141.2 Member Function Documentation . . . 556
12.141.2.1 enable_reed_solomon() . . . 556
12.141.2.2 get_A0G() . . . 556
12.141.2.3 get_A1G() . . . 556
12.141.2.4 get_flag_CRC_test() . . . 556
12.141.2.5 get_flag_GGTO() . . . 556
12.141.2.6 get_flag_TOW_set() . . . 556
12.141.2.7 get_Galileo_week() . . . 556
12.141.2.8 get_t0G() . . . 556
12.141.2.9 get_TOW0() . . . 556
12.141.2.10 get_TOW5() . . . 556
12.141.2.11 get_TOW6() . . . 557
12.141.2.12 get_WN0G() . . . 557
12.141.2.13 init_PRN() . . . 557
12.141.2.14 is_alert_page() . . . 557
12.141.2.15 is_TOW0_set() . . . 557
12.141.2.16 is_TOW5_set() . . . 557
12.141.2.17 is_TOW6_set() . . . 557
12.141.2.18 set_flag_TOW_set() . . . 557
12.141.2.19 set_TOW0_flag() . . . 557
12.141.2.20 set_TOW5_flag() . . . 557
12.141.2.21 set_TOW6_flag() . . . 557
12.142 Galileo_Iono Class Reference . . . 558
12.142.1 Detailed Description . . . 558
12.142.2 Constructor & Destructor Documentation . . . 558
12.142.2.1 Galileo_Iono() . . . 558
12.142.3 Member Function Documentation . . . 559
12.142.3.1 serialize() . . . 559
12.142.4 Member Data Documentation . . . 559
12.142.4.1 ai0 . . . 559
12.142.4.2 ai1 . . . 559
12.142.4.3 ai2 . . . 559
12.142.4.4 Region1_flag . . . 559
12.142.4.5 Region2_flag . . . 559
12.142.4.6 Region3_flag . . . 559

12.142.4.7 Region4_flag . . . 560
12.142.4.8 Region5_flag . . . 560
12.142.4.9 tow . . . 560
12.142.4.10 valid . . . 560
12.142.4.11 WN . . . 560
12.143 Galileo_ISM Class Reference . . . 560
12.143.1 Detailed Description . . . 561
12.143.2 Constructor & Destructor Documentation . . . 561
12.143.2.1 Galileo_ISM() . . . 561
12.144 galileo_pcps_8ms_acquisition_cc Class Reference . . . 561
12.144.1 Detailed Description . . . 562
12.144.2 Constructor & Destructor Documentation . . . 562
12.144.2.1 ~galileo_pcps_8ms_acquisition_cc() . . . 562
12.144.3 Member Function Documentation . . . 562
12.144.3.1 general_work() . . . 562
12.144.3.2 mag() . . . 562
12.144.3.3 set_active() . . . 562
12.144.3.4 set_channel() . . . 563
12.144.3.5 set_channel_fsm() . . . 563
12.144.3.6 set_gnss_synchro() . . . 563
12.144.3.7 set_local_code() . . . 563
12.145 Galileo_Reduced_CED Class Reference . . . 564
12.145.1 Detailed Description . . . 565
12.145.2 Constructor & Destructor Documentation . . . 565
12.145.2.1 Galileo_Reduced_CED() . . . 565
12.145.3 Member Function Documentation . . . 565
12.145.3.1 compute_eph() . . . 565
12.145.3.2 is_valid_at() . . . 565
12.145.3.3 reference_time() . . . 565
12.145.4 Member Data Documentation . . . 565
12.145.4.1 af0red . . . 565
12.145.4.2 af1red . . . 565
12.145.4.3 BGD_E1E5b . . . 565
12.145.4.4 CED_VALIDITY_SECONDS . . . 565
12.145.4.5 DeltaAred . . . 565
12.145.4.6 Deltai0red . . . 566
12.145.4.7 E1B_DVS . . . 566
12.145.4.8 E1B_HS . . . 566
12.145.4.9 E5b_DVS . . . 566
12.145.4.10 E5b_HS . . . 566
12.145.4.11 exred . . . 566
12.145.4.12 eyred . . . 566

12.145.4.13 lambda0red . . . 566
12.145.4.14 Omega0red . . . 566
12.145.4.15 PRN . . . 566
12.145.4.16 TOTRedCED . . . 567
12.145.4.17 WN . . . 567
12.146 galileo_telemetry_decoder_gs Class Reference . . . 567
12.146.1 Detailed Description . . . 568
12.146.2 Member Function Documentation . . . 568
12.146.2.1 general_work() . . . 568
12.146.2.2 reset() . . . 568
12.146.2.3 set_channel() . . . 568
12.146.2.4 set_satellite() . . . 568
12.147 galileo_tow_map Class Reference . . . 568
12.147.1 Detailed Description . . . 569
12.147.2 Constructor & Destructor Documentation . . . 569
12.147.2.1 ∼galileo_tow_map() . . . 569
12.148 Galileo_Utc_Model Class Reference . . . 569
12.148.1 Detailed Description . . . 570
12.148.2 Constructor & Destructor Documentation . . . 570
12.148.2.1 Galileo_Utc_Model() . . . 570
12.148.3 Member Function Documentation . . . 570
12.148.3.1 GST_to_UTC_time() . . . 570
12.148.3.2 serialize() . . . 570
12.148.3.3 truncated_week_diff() . . . 570
12.148.4 Member Data Documentation . . . 570
12.148.4.1 A0 . . . 570
12.148.4.2 A1 . . . 570
12.148.4.3 A_0G . . . 570
12.148.4.4 A_1G . . . 571
12.148.4.5 Delta_tLS . . . 571
12.148.4.6 Delta_tLSF . . . 571
12.148.4.7 DN . . . 571
12.148.4.8 flag_GGTO . . . 571
12.148.4.9 flag_utc_model . . . 571
12.148.4.10 t_0G . . . 571
12.148.4.11 tot . . . 571
12.148.4.12 WN_0G . . . 571
12.148.4.13 WN_LSF . . . 571
12.148.4.14 WNot . . . 571
12.149 GalileoE1TcpConnectorTracking Class Reference . . . 572
12.149.1 Detailed Description . . . 572
12.149.2 Member Function Documentation . . . 572

12.149.2.1 connect() . . . 572
12.149.2.2 disconnect() . . . 573
12.149.2.3 get_left_block() . . . 573
12.149.2.4 get_right_block() . . . 573
12.149.2.5 implementation() . . . 573
12.149.2.6 item_size() . . . 573
12.149.2.7 role() . . . 573
12.149.2.8 set_channel() . . . 573
12.149.2.9 set_gnss_synchro() . . . 573
12.149.2.10 start_tracking() . . . 573
12.149.2.11 stop_tracking() . . . 573
12.150 GalileoTowMapEntry Struct Reference . . . 574
12.150.1 Detailed Description . . . 574
12.150.2 Constructor & Destructor Documentation . . . 574
12.150.2.1 GalileoTowMapEntry() . . . 574
12.150.3 Member Data Documentation . . . 574
12.150.3.1 sample_counter . . . 574
12.150.3.2 tow_ms . . . 574
12.150.3.3 week . . . 574
12.151 GenSignalSource Class Reference . . . 574
12.151.1 Detailed Description . . . 575
12.151.2 Constructor & Destructor Documentation . . . 575
12.151.2.1 GenSignalSource() . . . 575
12.151.3 Member Function Documentation . . . 575
12.151.3.1 connect() . . . 575
12.151.3.2 disconnect() . . . 576
12.151.3.3 get_left_block() . . . 576
12.151.3.4 get_right_block() . . . 576
12.151.3.5 getRfChannels() . . . 576
12.151.3.6 implementation() . . . 576
12.151.3.7 item_size() . . . 576
12.151.3.8 role() . . . 576
12.151.3.9 signal_generator() . . . 576
12.152 Geohash Class Reference . . . 576
12.152.1 Detailed Description . . . 577
12.152.2 Member Function Documentation . . . 577
12.152.2.1 decode() . . . 577
12.152.2.2 encode() . . . 577
12.153 GeoJSON_Printer Class Reference . . . 577
12.153.1 Detailed Description . . . 578
12.154 geph_t Struct Reference . . . 578
12.154.1 Detailed Description . . . 578

12.154.2 Member Data Documentation . . . 578
12.154.2.1 acc . . . 578
12.154.2.2 age . . . 578
12.154.2.3 dtaun . . . 578
12.154.2.4 frq . . . 579
12.154.2.5 gamn . . . 579
12.154.2.6 iode . . . 579
12.154.2.7 pos . . . 579
12.154.2.8 sat . . . 579
12.154.2.9 sva . . . 579
12.154.2.10 svh . . . 579
12.154.2.11 taun . . . 579
12.154.2.12 toe . . . 579
12.154.2.13 tof . . . 579
12.154.2.14 vel . . . 579
12.155 Glonass_Gnav_Almanac Class Reference . . . 580
12.155.1 Detailed Description . . . 581
12.155.2 Constructor & Destructor Documentation . . . 581
12.155.2.1 Glonass_Gnav_Almanac() . . . 581
12.155.3 Member Function Documentation . . . 581
12.155.3.1 satellite_position() . . . 581
12.155.3.2 serialize() . . . 581
12.155.4 Member Data Documentation . . . 581
12.155.4.1 d_C_n . . . 581
12.155.4.2 d_Delta_i_n_A . . . 581
12.155.4.3 d_Delta_T_n_A . . . 582
12.155.4.4 d_Delta_T_n_A_dot . . . 582
12.155.4.5 d_epsilon_n_A . . . 582
12.155.4.6 d_H_n_A . . . 582
12.155.4.7 d_KP . . . 582
12.155.4.8 d_l_n . . . 582
12.155.4.9 d_lambda_n_A . . . 582
12.155.4.10 d_M_n_A . . . 582
12.155.4.11 d_N_4 . . . 583
12.155.4.12 d_N_A . . . 583
12.155.4.13 d_n_A . . . 583
12.155.4.14 d_omega_n_A . . . 583
12.155.4.15 d_t_lambda_n_A . . . 583
12.155.4.16 d_tau_n_A . . . 583
12.155.4.17 i_satellite_freq_channel . . . 583
12.155.4.18 i_satellite_slot_number . . . 583
12.155.4.19 PRN . . . 583

12.156 Glonass_Gnav_Ephemeris Class Reference . . . 584
12.156.1 Detailed Description . . . 586
12.156.2 Constructor & Destructor Documentation . . . 586
12.156.2.1 Glonass_Gnav_Ephemeris() . . . 586
12.156.3 Member Function Documentation . . . 586
12.156.3.1 compute_GLONASS_time() . . . 586
12.156.3.2 glot_to_gpst() . . . 586
12.156.3.3 glot_to_utc() . . . 587
12.156.3.4 serialize() . . . 587
12.156.3.5 sv_clock_drift() . . . 588
12.156.4 Member Data Documentation . . . 588
12.156.4.1 d_AXn . . . 588
12.156.4.2 d_AYn . . . 588
12.156.4.3 d_AZn . . . 588
12.156.4.4 d_B_n . . . 588
12.156.4.5 d_Delta_tau_n . . . 589
12.156.4.6 d_dtr . . . 589
12.156.4.7 d_E_n . . . 589
12.156.4.8 d_F_T . . . 589
12.156.4.9 d_gamma_n . . . 589
12.156.4.10 d_iode . . . 589
12.156.4.11 d_kp_leap_correction_s . . . 589
12.156.4.12 d_l3rd_n . . . 589
12.156.4.13 d_l5th_n . . . 590
12.156.4.14 d_M . . . 590
12.156.4.15 d_m . . . 590
12.156.4.16 d_n . . . 590
12.156.4.17 d_N_T . . . 590
12.156.4.18 d_P . . . 590
12.156.4.19 d_P_1 . . . 590
12.156.4.20 d_P_2 . . . 590
12.156.4.21 d_P_3 . . . 591
12.156.4.22 d_P_4 . . . 591
12.156.4.23 d_satClkDrift . . . 591
12.156.4.24 d_t_b . . . 591
12.156.4.25 d_t_k . . . 591
12.156.4.26 d_tau_c . . . 591
12.156.4.27 d_tau_n . . . 591
12.156.4.28 d_tod . . . 591
12.156.4.29 d_TOW . . . 591
12.156.4.30 d_VXn . . . 592
12.156.4.31 d_VYn . . . 592

12.156.4.32 d_VZn . . . 592
12.156.4.33 d_WN . . . 592
12.156.4.34 d_Xn . . . 592
12.156.4.35 d_Yn . . . 592
12.156.4.36 d_yr . . . 592
12.156.4.37 d_Zn . . . 592
12.156.4.38 i_satellite_freq_channel . . . 593
12.156.4.39 i_satellite_slot_number . . . 593
12.156.4.40 PRN . . . 593
12.157 Glonass_Gnav_Navigation_Message Class Reference . . . 593
12.157.1 Detailed Description . . . 594
12.157.2 Constructor & Destructor Documentation . . . 594
12.157.2.1 Glonass_Gnav_Navigation_Message() . . . 594
12.157.3 Member Function Documentation . . . 594
12.157.3.1 CRC_test() . . . 594
12.157.3.2 get_alm_satellite_slot_number() . . . 594
12.157.3.3 get_almanac() . . . 594
12.157.3.4 get_ephemeris() . . . 595
12.157.3.5 get_flag_CRC_test() . . . 595
12.157.3.6 get_flag_TOW_new() . . . 595
12.157.3.7 get_flag_update_slot_number() . . . 595
12.157.3.8 get_frame_number() . . . 595
12.157.3.9 get_utc_model() . . . 595
12.157.3.10 have_new_almanac() . . . 595
12.157.3.11 have_new_ephemeris() . . . 595
12.157.3.12 have_new_utc_model() . . . 595
12.157.3.13 is_flag_TOW_set() . . . 595
12.157.3.14 set_flag_ephemeris_str_1() . . . 596
12.157.3.15 set_flag_ephemeris_str_2() . . . 596
12.157.3.16 set_flag_ephemeris_str_3() . . . 596
12.157.3.17 set_flag_ephemeris_str_4() . . . 596
12.157.3.18 set_flag_TOW_new() . . . 596
12.157.3.19 set_flag_TOW_set() . . . 596
12.157.3.20 set_flag_update_slot_number() . . . 596
12.157.3.21 set_rf_link() . . . 596
12.157.3.22 string_decoder() . . . 596
12.158 glonass_gnav_telemetry_decoder_gs Class Reference . . . 597
12.158.1 Detailed Description . . . 598
12.158.2 Constructor & Destructor Documentation . . . 598
12.158.2.1 ~glonass_gnav_telemetry_decoder_gs() . . . 598
12.158.3 Member Function Documentation . . . 598
12.158.3.1 general_work() . . . 598

12.158.3.2 reset() . . . 598
12.158.3.3 set_channel() . . . 598
12.158.3.4 set_satellite() . . . 598
12.159 Glonass_Gnav_Utc_Model Class Reference . . . 598
12.159.1 Detailed Description . . . 599
12.159.2 Constructor & Destructor Documentation . . . 599
12.159.2.1 Glonass_Gnav_Utc_Model() . . . 599
12.159.3 Member Function Documentation . . . 600
12.159.3.1 announced_leap_second() . . . 600
12.159.3.2 serialize() . . . 600
12.159.3.3 utc_time() . . . 600
12.159.4 Member Data Documentation . . . 600
12.159.4.1 d_B1 . . . 600
12.159.4.2 d_B2 . . . 600
12.159.4.3 d_KP . . . 600
12.159.4.4 d_N_4 . . . 600
12.159.4.5 d_N_A . . . 601
12.159.4.6 d_tau_c . . . 601
12.159.4.7 d_tau_gps . . . 601
12.159.4.8 valid . . . 601
12.160 GnMaxSignalSource Class Reference . . . 601
12.160.1 Detailed Description . . . 602
12.160.2 Member Function Documentation . . . 602
12.160.2.1 connect() . . . 602
12.160.2.2 disconnect() . . . 602
12.160.2.3 get_left_block() . . . 602
12.160.2.4 get_right_block() . . . 602
12.160.2.5 item_size() . . . 602
12.161 Gnss_Almanac Class Reference . . . 603
12.161.1 Detailed Description . . . 604
12.161.2 Constructor & Destructor Documentation . . . 604
12.161.2.1 Gnss_Almanac() . . . 604
12.161.3 Member Function Documentation . . . 604
12.161.3.1 predicted_doppler() . . . 604
12.161.3.2 satellitePosVelComputation() . . . 605
12.161.4 Member Data Documentation . . . 605
12.161.4.1 af0 . . . 605
12.161.4.2 af1 . . . 605
12.161.4.3 delta_i . . . 605
12.161.4.4 ecc . . . 605
12.161.4.5 from_startup_load . . . 605
12.161.4.6 M_0 . . . 605

12.161.4.7 omega . . . 605
12.161.4.8 OMEGA_0 . . . 605
12.161.4.9 OMEGAdot . . . 606
12.161.4.10 PRN . . . 606
12.161.4.11 sqrtA . . . 606
12.161.4.12 System . . . 606
12.161.4.13 toa . . . 606
12.161.4.14 WNa . . . 606
12.162 Gnss_circular_deque< T > Class Template Reference . . . 606
12.162.1 Detailed Description . . . 607
12.163 Gnss_Crypto Class Reference . . . 607
12.163.1 Detailed Description . . . 608
12.163.2 Constructor & Destructor Documentation . . . 608
12.163.2.1 Gnss_Crypto() [1/2] . . . 608
12.163.2.2 Gnss_Crypto() [2/2] . . . 608
12.163.2.3 ~Gnss_Crypto() . . . 608
12.163.3 Member Function Documentation . . . 608
12.163.3.1 clear_public_key() . . . 608
12.163.3.2 compute_CMAC_AES() . . . 608
12.163.3.3 compute_HMAC_SHA_256() . . . 609
12.163.3.4 compute_SHA3_256() . . . 609
12.163.3.5 compute_SHA_256() . . . 609
12.163.3.6 get_merkle_root() . . . 609
12.163.3.7 get_public_key_compressed() . . . 609
12.163.3.8 get_public_key_type() . . . 609
12.163.3.9 have_public_key() . . . 609
12.163.3.10 set_merkle_root() . . . 609
12.163.3.11 set_public_key() . . . 610
12.163.3.12 set_public_key_type() . . . 610
12.163.3.13 store_public_key() . . . 610
12.163.3.14 verify_signature_ecdsa_p256() . . . 610
12.163.3.15 verify_signature_ecdsa_p521() . . . 610
12.164 Gnss_Ephemeris Class Reference . . . 610
12.164.1 Detailed Description . . . 612
12.164.2 Member Function Documentation . . . 612
12.164.2.1 predicted_doppler() . . . 612
12.164.2.2 satellitePosition() . . . 613
12.164.2.3 sv_clock_drift() . . . 613
12.164.3 Member Data Documentation . . . 613
12.164.3.1 A0 . . . 613
12.164.3.2 Adot . . . 613
12.164.3.3 af0 . . . 613

12.164.3.4 af1 . . . 614
12.164.3.5 af2 . . . 614
12.164.3.6 Cic . . . 614
12.164.3.7 Cis . . . 614
12.164.3.8 Crc . . . 614
12.164.3.9 Crs . . . 614
12.164.3.10 Cuc . . . 614
12.164.3.11 Cus . . . 615
12.164.3.12 delta_n . . . 615
12.164.3.13 delta_ndot . . . 615
12.164.3.14 dtr . . . 615
12.164.3.15 ecc . . . 615
12.164.3.16 i_0 . . . 615
12.164.3.17 idot . . . 615
12.164.3.18 M_0 . . . 615
12.164.3.19 omega . . . 616
12.164.3.20 OMEGA_0 . . . 616
12.164.3.21 OMEGAdot . . . 616
12.164.3.22 PRN . . . 616
12.164.3.23 satClkDrift . . . 616
12.164.3.24 satpos_X . . . 616
12.164.3.25 satpos_Y . . . 616
12.164.3.26 satpos_Z . . . 617
12.164.3.27 satvel_X . . . 617
12.164.3.28 satvel_Y . . . 617
12.164.3.29 satvel_Z . . . 617
12.164.3.30 sqrtA . . . 617
12.164.3.31 System . . . 617
12.164.3.32 toc . . . 617
12.164.3.33 toe . . . 617
12.164.3.34 tow . . . 617
12.164.3.35 WN . . . 618
12.165 Gnss_Satellite Class Reference . . . 618
12.165.1 Detailed Description . . . 619
12.165.2 Constructor & Destructor Documentation . . . 619
12.165.2.1 Gnss_Satellite() [1/4] . . . 619
12.165.2.2 Gnss_Satellite() [2/4] . . . 619
12.165.2.3 ~Gnss_Satellite() . . . 619
12.165.2.4 Gnss_Satellite() [3/4] . . . 619
12.165.2.5 Gnss_Satellite() [4/4] . . . 619
12.165.3 Member Function Documentation . . . 619
12.165.3.1 get_block() . . . 619

12.165.3.2 get_PRN() . . . 619
12.165.3.3 get_rf_link() . . . 620
12.165.3.4 get_system() . . . 620
12.165.3.5 get_system_short() . . . 620
12.165.3.6 operator=() [1/2] . . . 620
12.165.3.7 operator=() [2/2] . . . 620
12.165.3.8 update_PRN() . . . 620
12.165.3.9 what_block() . . . 620
12.165.4 Friends And Related Symbol Documentation . . . 620
12.165.4.1 operator<< . . . 620
12.165.4.2 operator== . . . 620
12.166 gnss_sdr_fpga_sample_counter Class Reference . . . 621
12.166.1 Detailed Description . . . 621
12.167 gnss_sdr_sample_counter Class Reference . . . 621
12.167.1 Detailed Description . . . 621
12.168 Gnss_Sdr_Supl_Client Class Reference . . . 622
12.168.1 Detailed Description . . . 623
12.168.2 Member Function Documentation . . . 623
12.168.2.1 load_cnav_ephemeris_xml() . . . 623
12.168.2.2 load_cnav_utc_xml() . . . 624
12.168.2.3 load_ephemeris_xml() . . . 624
12.168.2.4 load_gal_almanac_xml() . . . 624
12.168.2.5 load_gal_ephemeris_xml() . . . 624
12.168.2.6 load_gal_iono_xml() . . . 624
12.168.2.7 load_gal_utc_xml() . . . 624
12.168.2.8 load_glo_utc_xml() . . . 624
12.168.2.9 load_gnav_ephemeris_xml() . . . 624
12.168.2.10 load_gps_almanac_xml() . . . 624
12.168.2.11 load_iono_xml() . . . 624
12.168.2.12 load_ref_location_xml() . . . 625
12.168.2.13 load_ref_time_xml() . . . 625
12.168.2.14 load_utc_xml() . . . 625
12.168.2.15 save_cnav_ephemeris_map_xml() . . . 625
12.168.2.16 save_cnav_utc_xml() . . . 625
12.168.2.17 save_ephemeris_map_xml() . . . 625
12.168.2.18 save_gal_almanac_xml() . . . 625
12.168.2.19 save_gal_ephemeris_map_xml() . . . 625
12.168.2.20 save_gal_iono_xml() . . . 625
12.168.2.21 save_gal_utc_xml() . . . 626
12.168.2.22 save_glo_utc_xml() . . . 626
12.168.2.23 save_gnav_ephemeris_map_xml() . . . 626
12.168.2.24 save_gps_almanac_xml() . . . 626

12.168.2.25 save_iono_xml() . . . 626
12.168.2.26 save_ref_location_xml() . . . 626
12.168.2.27 save_ref_time_xml() . . . 626
12.168.2.28 save_utc_xml() . . . 626
12.168.3 Member Data Documentation . . . 627
12.168.3.1 gal_almanac_map . . . 627
12.168.3.2 gal_ephemeris_map . . . 627
12.168.3.3 gal_iono . . . 627
12.168.3.4 gal_utc . . . 627
12.168.3.5 glo_gnav_utc . . . 627
12.168.3.6 glonass_gnav_ephemeris_map . . . 627
12.168.3.7 gps_acq_map . . . 627
12.168.3.8 gps_almanac_map . . . 627
12.168.3.9 gps_cnav_ephemeris_map . . . 627
12.168.3.10 gps_cnav_utc . . . 627
12.168.3.11 gps_ephemeris_map . . . 627
12.168.3.12 gps_iono . . . 627
12.168.3.13 gps_ref_loc . . . 628
12.168.3.14 gps_time . . . 628
12.168.3.15 gps_utc . . . 628
12.168.3.16 request . . . 628
12.168.3.17 server_name . . . 628
12.168.3.18 server_port . . . 628
12.169 gnss_sdr_time_counter Class Reference . . . 628
12.169.1 Detailed Description . . . 628
12.170 Gnss_Sdr_Timestamp Class Reference . . . 629
12.170.1 Detailed Description . . . 629
12.171 Gnss_Sdr_Valve Class Reference . . . 629
12.171.1 Detailed Description . . . 630
12.172 Gnss_Signal Class Reference . . . 630
12.172.1 Detailed Description . . . 630
12.172.2 Member Function Documentation . . . 630
12.172.2.1 get_satellite() . . . 630
12.172.2.2 get_signal_str() . . . 630
12.172.3 Friends And Related Symbol Documentation . . . 631
12.172.3.1 operator<< . . . 631
12.172.3.2 operator== . . . 631
12.173 Gnss_Synchro Class Reference . . . 631
12.173.1 Detailed Description . . . 633
12.173.2 Constructor & Destructor Documentation . . . 633
12.173.2.1 Gnss_Synchro() [1/3] . . . 633
12.173.2.2 ~Gnss_Synchro() . . . 633

12.173.2.3 Gnss_Synchro() [2/3] . . . 633
12.173.2.4 Gnss_Synchro() [3/3] . . . 633
12.173.3 Member Function Documentation . . . 633
12.173.3.1 operator=() [1/2] . . . 633
12.173.3.2 operator=() [2/2] . . . 633
12.173.3.3 serialize() . . . 633
12.173.4 Member Data Documentation . . . 634
12.173.4.1 Acq_delay_samples . . . 634
12.173.4.2 Acq_doppler_hz . . . 634
12.173.4.3 Acq_doppler_step . . . 634
12.173.4.4 Acq_samplestamp_samples . . . 634
12.173.4.5 Carrier_Doppler_hz . . . 634
12.173.4.6 Carrier_phase_rads . . . 634
12.173.4.7 Channel_ID . . . 634
12.173.4.8 CN0_dB_hz . . . 634
12.173.4.9 Code_phase_samples . . . 635
12.173.4.10 correlation_length_ms . . . 635
12.173.4.11 Flag_carrier_phase_continuous . . . 635
12.173.4.12 Flag_cycle_slip . . . 635
12.173.4.13 Flag_half_cycle_slip . . . 635
12.173.4.14 Flag_PLL_180_deg_phase_locked . . . 635
12.173.4.15 Flag_valid_acquisition . . . 635
12.173.4.16 Flag_valid_pseudorange . . . 635
12.173.4.17 Flag_valid_symbol_output . . . 636
12.173.4.18 Flag_valid_word . . . 636
12.173.4.19 fs . . . 636
12.173.4.20 interp_TOW_ms . . . 636
12.173.4.21 PRN . . . 636
12.173.4.22 Prompt_I . . . 636
12.173.4.23 Prompt_Q . . . 636
12.173.4.24 Pseudorange_m . . . 636
12.173.4.25 RX_time . . . 636
12.173.4.26 Signal . . . 637
12.173.4.27 System . . . 637
12.173.4.28 TOW_at_current_symbol_ms . . . 637
12.173.4.29 Tracking_sample_counter . . . 637
12.174 gnss_synchro_monitor Class Reference . . . 637
12.174.1 Detailed Description . . . 638
12.174.2 Constructor & Destructor Documentation . . . 638
12.174.2.1 ~gnss_synchro_monitor() . . . 638
12.175 Gnss_Synchro_Udp_Sink Class Reference . . . 638
12.175.1 Detailed Description . . . 638

12.176 GNSSBlockInterface Class Reference . . . 638
12.176.1 Detailed Description . . . 639
12.176.2 Member Function Documentation . . . 639
12.176.2.1 connect() . . . 639
12.176.2.2 disconnect() . . . 640
12.176.2.3 get_left_block() [1/2] . . . 640
12.176.2.4 get_left_block() [2/2] . . . 640
12.176.2.5 get_right_block() [1/2] . . . 640
12.176.2.6 get_right_block() [2/2] . . . 640
12.176.2.7 implementation() . . . 640
12.176.2.8 item_size() . . . 640
12.176.2.9 role() . . . 640
12.176.2.10 start() . . . 640
12.177 GNSSFlowgraph Class Reference . . . 641
12.177.1 Detailed Description . . . 641
12.177.2 Constructor & Destructor Documentation . . . 642
12.177.2.1 GNSSFlowgraph() . . . 642
12.177.2.2 ~GNSSFlowgraph() . . . 642
12.177.3 Member Function Documentation . . . 642
12.177.3.1 acquisition_manager() . . . 642
12.177.3.2 apply_action() . . . 642
12.177.3.3 connect() . . . 642
12.177.3.4 connected() . . . 642
12.177.3.5 disconnect() . . . 642
12.177.3.6 get_pvt() . . . 643
12.177.3.7 MaybeUpdateVisibility() . . . 643
12.177.3.8 priorize_satellites() . . . 643
12.177.3.9 running() . . . 643
12.177.3.10 send_telemetry_msg() . . . 643
12.177.3.11 set_configuration() . . . 643
12.177.3.12 start() . . . 643
12.177.3.13 stop() . . . 643
12.177.3.14 stop_duplicated_satellite_channels() . . . 643
12.177.3.15 UpdateVisibilityReference() . . . 644
12.177.3.16 visibility_aware_search_enabled() . . . 644
12.177.3.17 wait() . . . 644
12.178 GnssSdrLogSink Class Reference . . . 644
12.178.1 Detailed Description . . . 644
12.178.2 Member Function Documentation . . . 644
12.178.2.1 filename() . . . 644
12.179 GnssSdrLogSinkGuard Class Reference . . . 645
12.179.1 Detailed Description . . . 645

12.180 GnssTime Class Reference . . . 645
12.180.1 Detailed Description . . . 645
12.180.2 Member Data Documentation . . . 645
12.180.2.1 rx_time . . . 645
12.180.2.2 tow_ms . . . 645
12.180.2.3 tow_ms_fraction . . . 645
12.180.2.4 week . . . 645
12.181 Gps_Acq_Assist Class Reference . . . 646
12.181.1 Detailed Description . . . 646
12.181.2 Constructor & Destructor Documentation . . . 646
12.181.2.1 Gps_Acq_Assist() . . . 646
12.181.3 Member Data Documentation . . . 646
12.181.3.1 Azimuth . . . 646
12.181.3.2 Code_Phase . . . 647
12.181.3.3 Code_Phase_int . . . 647
12.181.3.4 Code_Phase_window . . . 647
12.181.3.5 Doppler0 . . . 647
12.181.3.6 Doppler1 . . . 647
12.181.3.7 dopplerUncertainty . . . 647
12.181.3.8 Elevation . . . 647
12.181.3.9 GPS_Bit_Number . . . 647
12.181.3.10 PRN . . . 647
12.181.3.11 tow . . . 647
12.182 Gps_Almanac Class Reference . . . 648
12.182.1 Detailed Description . . . 649
12.182.2 Constructor & Destructor Documentation . . . 649
12.182.2.1 Gps_Almanac() . . . 649
12.182.3 Member Function Documentation . . . 649
12.182.3.1 get_system() . . . 649
12.182.3.2 serialize() . . . 649
12.182.3.3 set_system() . . . 649
12.182.4 Member Data Documentation . . . 650
12.182.4.1 AS_status . . . 650
12.182.4.2 SV_health . . . 650
12.183 Gps_CNAV_Eop Class Reference . . . 650
12.183.1 Detailed Description . . . 650
12.183.2 Member Data Documentation . . . 651
12.183.2.1 delta_ut1_gps . . . 651
12.183.2.2 delta_ut1_gps_dot . . . 651
12.183.2.3 pm_x . . . 651
12.183.2.4 pm_x_dot . . . 651
12.183.2.5 pm_y . . . 651

12.183.2.6 pm_y_dot . . . 651
12.183.2.7 PRN . . . 651
12.183.2.8 t_eop . . . 651
12.183.2.9 tow . . . 651
12.183.2.10 valid . . . 651
12.184 Gps_CNAV_Ephemeris Class Reference . . . 652
12.184.1 Detailed Description . . . 655
12.184.2 Constructor & Destructor Documentation . . . 655
12.184.2.1 Gps_CNAV_Ephemeris() . . . 655
12.184.3 Member Function Documentation . . . 655
12.184.3.1 serialize() . . . 655
12.184.4 Member Data Documentation . . . 655
12.184.4.1 alert_flag . . . 655
12.184.4.2 antispoofing_flag . . . 655
12.184.4.3 delta_A . . . 656
12.184.4.4 delta_OMEGAdot . . . 656
12.184.4.5 integrity_status_flag . . . 656
12.184.4.6 ISCL1 . . . 656
12.184.4.7 ISCL2 . . . 656
12.184.4.8 ISCL5I . . . 656
12.184.4.9 ISCL5Q . . . 656
12.184.4.10 l2c_phasing_flag . . . 656
12.184.4.11 signal_health . . . 656
12.184.4.12 TGD . . . 657
12.184.4.13 toe1 . . . 657
12.184.4.14 toe2 . . . 657
12.184.4.15 top . . . 657
12.184.4.16 URA . . . 657
12.184.4.17 URA0 . . . 657
12.184.4.18 URA1 . . . 657
12.184.4.19 URA2 . . . 657
12.184.4.20 URAED . . . 657
12.184.4.21 URANED0 . . . 658
12.184.4.22 URANED1 . . . 658
12.184.4.23 URANED2 . . . 658
12.184.4.24 WNop . . . 658
12.185 Gps_CNAV_Iono Class Reference . . . 658
12.185.1 Detailed Description . . . 659
12.185.2 Constructor & Destructor Documentation . . . 659
12.185.2.1 Gps_CNAV_Iono() . . . 659
12.186 Gps_CNAV_Navigation_Message Class Reference . . . 659
12.186.1 Detailed Description . . . 660

12.186.2 Constructor & Destructor Documentation . . . 660
12.186.2.1 Gps_CNAV_Navigation_Message() . . . 660
12.186.3 Member Function Documentation . . . 660
12.186.3.1 get_eop() . . . 660
12.186.3.2 get_ephemeris() . . . 660
12.186.3.3 get_iono() . . . 660
12.186.3.4 get_utc_model() . . . 660
12.186.3.5 have_new_eop() . . . 661
12.186.3.6 have_new_ephemeris() . . . 661
12.186.3.7 have_new_iono() . . . 661
12.186.3.8 have_new_utc_model() . . . 661
12.187 Gps_CNAV_Utc_Model Class Reference . . . 661
12.187.1 Detailed Description . . . 662
12.187.2 Constructor & Destructor Documentation . . . 662
12.187.2.1 Gps_CNAV_Utc_Model() . . . 662
12.188 Gps_Ephemeris Class Reference . . . 662
12.188.1 Detailed Description . . . 665
12.188.2 Constructor & Destructor Documentation . . . 665
12.188.2.1 Gps_Ephemeris() . . . 665
12.188.3 Member Function Documentation . . . 665
12.188.3.1 get_system() . . . 665
12.188.3.2 serialize() . . . 666
12.188.3.3 set_system() . . . 666
12.188.4 Member Data Documentation . . . 666
12.188.4.1 alert_flag . . . 666
12.188.4.2 antispoofing_flag . . . 666
12.188.4.3 AODO . . . 666
12.188.4.4 code_on_L2 . . . 666
12.188.4.5 fit_interval_flag . . . 666
12.188.4.6 integrity_status_flag . . . 667
12.188.4.7 IODC . . . 667
12.188.4.8 IODE_SF2 . . . 667
12.188.4.9 IODE_SF3 . . . 667
12.188.4.10 L2_P_data_flag . . . 667
12.188.4.11 satelliteBlock . . . 667
12.188.4.12 spare1 . . . 667
12.188.4.13 spare2 . . . 667
12.188.4.14 SV_accuracy . . . 668
12.188.4.15 SV_health . . . 668
12.188.4.16 TGD . . . 668
12.189 Gps_Iono Class Reference . . . 668
12.189.1 Detailed Description . . . 669

12.189.2 Constructor & Destructor Documentation . . . 669
12.189.2.1 Gps_Iono() . . . 669
12.189.3 Member Function Documentation . . . 669
12.189.3.1 serialize() . . . 669
12.189.4 Member Data Documentation . . . 669
12.189.4.1 alpha0 . . . 669
12.189.4.2 alpha1 . . . 669
12.189.4.3 alpha2 . . . 670
12.189.4.4 alpha3 . . . 670
12.189.4.5 beta0 . . . 670
12.189.4.6 beta1 . . . 670
12.189.4.7 beta2 . . . 670
12.189.4.8 beta3 . . . 670
12.189.4.9 valid . . . 670
12.190 Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc Class Reference . . . 670
12.190.1 Detailed Description . . . 671
12.191 Gps_L1_Ca_Gaussian_Tracking_cc Class Reference . . . 671
12.191.1 Detailed Description . . . 672
12.192 Gps_L1_Ca_Tcp_Connector_Tracking_cc Class Reference . . . 672
12.192.1 Detailed Description . . . 672
12.193 gps_l1_ca_telemetry_decoder_gs Class Reference . . . 672
12.193.1 Detailed Description . . . 673
12.193.2 Member Function Documentation . . . 673
12.193.2.1 general_work() . . . 673
12.193.2.2 reset() . . . 673
12.193.2.3 set_channel() . . . 673
12.193.2.4 set_satellite() . . . 674
12.194 gps_l2c_telemetry_decoder_gs Class Reference . . . 674
12.194.1 Detailed Description . . . 675
12.194.2 Member Function Documentation . . . 675
12.194.2.1 general_work() . . . 675
12.194.2.2 reset() . . . 675
12.194.2.3 set_channel() . . . 675
12.194.2.4 set_satellite() . . . 675
12.195 gps_l5_telemetry_decoder_gs Class Reference . . . 675
12.195.1 Detailed Description . . . 676
12.195.2 Member Function Documentation . . . 676
12.195.2.1 reset() . . . 676
12.195.2.2 set_channel() . . . 676
12.195.2.3 set_satellite() . . . 676
12.196 Gps_Navigation_Message Class Reference . . . 676
12.196.1 Detailed Description . . . 677

12.196.2 Constructor & Destructor Documentation . . . 677
12.196.2.1 Gps_Navigation_Message() . . . 677
12.196.3 Member Function Documentation . . . 677
12.196.3.1 get_almanac() . . . 677
12.196.3.2 get_almanac_config_code() . . . 678
12.196.3.3 get_almanac_health() . . . 678
12.196.3.4 get_ephemeris() . . . 678
12.196.3.5 get_flag_iono_valid() . . . 678
12.196.3.6 get_flag_utc_model_valid() . . . 678
12.196.3.7 get_GPS_week() . . . 678
12.196.3.8 get_iono() . . . 678
12.196.3.9 get_satellite_PRN() . . . 678
12.196.3.10 get_TOW() . . . 678
12.196.3.11 get_utc_model() . . . 678
12.196.3.12 set_channel() . . . 679
12.196.3.13 set_satellite_PRN() . . . 679
12.196.3.14 subframe_decoder() . . . 679
12.196.3.15 utc_time() . . . 679
12.197 Gps_Utc_Model Class Reference . . . 679
12.197.1 Detailed Description . . . 680
12.197.2 Constructor & Destructor Documentation . . . 680
12.197.2.1 Gps_Utc_Model() . . . 680
12.197.3 Member Function Documentation . . . 680
12.197.3.1 serialize() . . . 680
12.197.4 Member Data Documentation . . . 680
12.197.4.1 A0 . . . 680
12.197.4.2 A1 . . . 680
12.197.4.3 A2 . . . 680
12.197.4.4 DeltaT_LS . . . 681
12.197.4.5 DeltaT_LSF . . . 681
12.197.4.6 DN . . . 681
12.197.4.7 tot . . . 681
12.197.4.8 valid . . . 681
12.197.4.9 WN_LSF . . . 681
12.197.4.10 WN_T . . . 681
12.198 GpsL1CaDllPllTrackingGPU Class Reference . . . 681
12.198.1 Detailed Description . . . 682
12.198.2 Member Function Documentation . . . 682
12.198.2.1 connect() . . . 682
12.198.2.2 disconnect() . . . 682
12.198.2.3 get_left_block() . . . 682
12.198.2.4 get_right_block() . . . 682

12.198.2.5 implementation() . . . 683
12.198.2.6 item_size() . . . 683
12.198.2.7 role() . . . 683
12.198.2.8 set_channel() . . . 683
12.198.2.9 set_gnss_synchro() . . . 683
12.198.2.10 start_tracking() . . . 683
12.198.2.11 stop_tracking() . . . 683
12.199 GpsL1CaGaussianTracking Class Reference . . . 683
12.199.1 Detailed Description . . . 684
12.199.2 Member Function Documentation . . . 684
12.199.2.1 connect() . . . 684
12.199.2.2 disconnect() . . . 684
12.199.2.3 get_left_block() . . . 685
12.199.2.4 get_right_block() . . . 685
12.199.2.5 implementation() . . . 685
12.199.2.6 item_size() . . . 685
12.199.2.7 role() . . . 685
12.199.2.8 set_channel() . . . 685
12.199.2.9 set_gnss_synchro() . . . 685
12.199.2.10 start_tracking() . . . 685
12.199.2.11 stop_tracking() . . . 685
12.200 GpsL1CaKfTracking Class Reference . . . 686
12.200.1 Detailed Description . . . 686
12.200.2 Member Function Documentation . . . 686
12.200.2.1 connect() . . . 686
12.200.2.2 disconnect() . . . 687
12.200.2.3 get_left_block() . . . 687
12.200.2.4 get_right_block() . . . 687
12.200.2.5 implementation() . . . 687
12.200.2.6 item_size() . . . 687
12.200.2.7 role() . . . 687
12.200.2.8 set_channel() . . . 687
12.200.2.9 set_gnss_synchro() . . . 687
12.200.2.10 start_tracking() . . . 687
12.200.2.11 stop_tracking() . . . 687
12.201 GpsL1CaTcpConnectorTracking Class Reference . . . 688
12.201.1 Detailed Description . . . 688
12.201.2 Member Function Documentation . . . 689
12.201.2.1 connect() . . . 689
12.201.2.2 disconnect() . . . 689
12.201.2.3 get_left_block() . . . 689
12.201.2.4 get_right_block() . . . 689

12.201.2.5 implementation() . . . 689
12.201.2.6 item_size() . . . 689
12.201.2.7 role() . . . 689
12.201.2.8 set_channel() . . . 689
12.201.2.9 set_gnss_synchro() . . . 689
12.201.2.10 start_tracking() . . . 690
12.201.2.11 stop_tracking() . . . 690
12.202 GPU_Complex Struct Reference . . . 690
12.202.1 Detailed Description . . . 690
12.202.2 Constructor & Destructor Documentation . . . 690
12.202.2.1 GPU_Complex() [1/2] . . . 690
12.202.2.2 GPU_Complex() [2/2] . . . 690
12.202.3 Member Function Documentation . . . 690
12.202.3.1 magnitude2() . . . 690
12.202.3.2 multiply_acc() . . . 691
12.202.3.3 operator∗() . . . 691
12.202.3.4 operator+() . . . 691
12.202.3.5 operator+=() . . . 691
12.202.4 Member Data Documentation . . . 691
12.202.4.1 i . . . 691
12.202.4.2 r . . . 691
12.203 GPU_Complex_Short Struct Reference . . . 691
12.203.1 Detailed Description . . . 691
12.203.2 Constructor & Destructor Documentation . . . 692
12.203.2.1 GPU_Complex_Short() . . . 692
12.203.3 Member Function Documentation . . . 692
12.203.3.1 magnitude2() . . . 692
12.203.3.2 operator∗() . . . 692
12.203.3.3 operator+() . . . 692
12.203.4 Member Data Documentation . . . 692
12.203.4.1 i . . . 692
12.203.4.2 r . . . 692
12.204 Gpx_Printer Class Reference . . . 692
12.204.1 Detailed Description . . . 693
12.205 Gr_Complex_Ip_Packet_Source Class Reference . . . 693
12.205.1 Detailed Description . . . 693
12.205.2 Member Typedef Documentation . . . 693
12.205.2.1 sptr . . . 693
12.206 GSS6450FileSignalSource Class Reference . . . 693
12.206.1 Detailed Description . . . 694
12.206.2 Member Function Documentation . . . 695
12.206.2.1 connect() . . . 695

12.206.2.2 disconnect() . . . 695
12.206.2.3 filename() . . . 695
12.206.2.4 get_left_block() . . . 695
12.206.2.5 get_right_block() [1/2] . . . 695
12.206.2.6 get_right_block() [2/2] . . . 695
12.206.2.7 getRfChannels() . . . 695
12.206.2.8 item_size() . . . 695
12.206.2.9 item_type() . . . 695
12.206.2.10 repeat() . . . 695
12.206.2.11 samples() . . . 695
12.206.2.12 sampling_frequency() . . . 696
12.207 gtime_t Struct Reference . . . 696
12.207.1 Detailed Description . . . 696
12.207.2 Member Data Documentation . . . 696
12.207.2.1 sec . . . 696
12.207.2.2 time . . . 696
12.208 half_cyc_tag Struct Reference . . . 696
12.208.1 Detailed Description . . . 696
12.208.2 Member Data Documentation . . . 696
12.208.2.1 corr . . . 696
12.208.2.2 freq . . . 697
12.208.2.3 next . . . 697
12.208.2.4 sat . . . 697
12.208.2.5 te . . . 697
12.208.2.6 ts . . . 697
12.208.2.7 valid . . . 697
12.209 HAS_bias_corrections Class Reference . . . 697
12.209.1 Detailed Description . . . 697
12.209.2 Member Data Documentation . . . 697
12.209.2.1 bias . . . 697
12.209.2.2 iod_set_id . . . 697
12.209.2.3 mask_id . . . 698
12.209.2.4 valid_until . . . 698
12.210 HAS_clock_corrections Class Reference . . . 698
12.210.1 Detailed Description . . . 698
12.210.2 Member Data Documentation . . . 698
12.210.2.1 clock_correction_m . . . 698
12.210.2.2 iod . . . 698
12.210.2.3 iod_set_id . . . 698
12.210.2.4 mask_id . . . 698
12.210.2.5 valid_until . . . 698
12.211 HAS_obs_corrections Class Reference . . . 699

12.211.1 Detailed Description . . . 699
12.211.2 Member Data Documentation . . . 699
12.211.2.1 code_bias_m . . . 699
12.211.2.2 phase_bias_cycle . . . 699
12.211.2.3 phase_bias_discontinuity . . . 699
12.211.2.4 phase_discontinuity_indicator . . . 699
12.211.2.5 signal . . . 699
12.212 HAS_orbit_corrections Class Reference . . . 699
12.212.1 Detailed Description . . . 699
12.212.2 Member Data Documentation . . . 700
12.212.2.1 cross_track_m . . . 700
12.212.2.2 in_track_m . . . 700
12.212.2.3 iod . . . 700
12.212.2.4 iod_set_id . . . 700
12.212.2.5 mask_id . . . 700
12.212.2.6 radial_m . . . 700
12.212.2.7 valid_until . . . 700
12.213 Has_Simple_Printer Class Reference . . . 700
12.213.1 Detailed Description . . . 700
12.214 std::hash< SensorIdentifier::value_type > Struct Reference . . . 701
12.214.1 Detailed Description . . . 701
12.214.2 Member Function Documentation . . . 701
12.214.2.1 operator()() . . . 701
12.215 HistogramBitSynchronizer Class Reference . . . 701
12.215.1 Detailed Description . . . 702
12.215.2 Constructor & Destructor Documentation . . . 702
12.215.2.1 HistogramBitSynchronizer() . . . 702
12.215.3 Member Function Documentation . . . 702
12.215.3.1 bins() . . . 702
12.215.3.2 edge_phase() . . . 702
12.215.3.3 epochs_until_next_edge() . . . 702
12.215.3.4 get_epoch_count() . . . 703
12.215.3.5 get_histogram() . . . 703
12.215.3.6 get_total_events() . . . 703
12.215.3.7 is_edge_epoch() . . . 703
12.215.3.8 locked() . . . 704
12.215.3.9 reset() . . . 704
12.215.3.10 update() . . . 704
12.216 hybrid_observables_gs Class Reference . . . 704
12.216.1 Detailed Description . . . 705
12.217 HybridObservables Class Reference . . . 705
12.217.1 Detailed Description . . . 706

12.217.2 Member Function Documentation . . . 706
12.217.2.1 connect() . . . 706
12.217.2.2 disconnect() . . . 706
12.217.2.3 get_left_block() . . . 706
12.217.2.4 get_right_block() . . . 706
12.217.2.5 implementation() . . . 706
12.217.2.6 item_size() . . . 706
12.217.2.7 reset() . . . 706
12.217.2.8 role() . . . 707
12.218 IbyteToCbyte Class Reference . . . 707
12.218.1 Detailed Description . . . 707
12.218.2 Member Function Documentation . . . 707
12.218.2.1 connect() . . . 707
12.218.2.2 disconnect() . . . 707
12.218.2.3 get_left_block() . . . 708
12.218.2.4 get_right_block() . . . 708
12.218.2.5 implementation() . . . 708
12.218.2.6 item_size() . . . 708
12.218.2.7 role() . . . 708
12.219 IbyteToComplex Class Reference . . . 708
12.219.1 Detailed Description . . . 709
12.219.2 Member Function Documentation . . . 709
12.219.2.1 connect() . . . 709
12.219.2.2 disconnect() . . . 709
12.219.2.3 get_left_block() . . . 709
12.219.2.4 get_right_block() . . . 709
12.219.2.5 implementation() . . . 709
12.219.2.6 item_size() . . . 709
12.219.2.7 role() . . . 709
12.220 IbyteToCshort Class Reference . . . 710
12.220.1 Detailed Description . . . 710
12.220.2 Member Function Documentation . . . 710
12.220.2.1 connect() . . . 710
12.220.2.2 disconnect() . . . 710
12.220.2.3 get_left_block() . . . 710
12.220.2.4 get_right_block() . . . 711
12.220.2.5 implementation() . . . 711
12.220.2.6 item_size() . . . 711
12.220.2.7 role() . . . 711
12.221 INIReader Class Reference . . . 711
12.221.1 Detailed Description . . . 711
12.221.2 Constructor & Destructor Documentation . . . 712

12.221.2.1 INIReader() . . . 712
12.221.3 Member Function Documentation . . . 712
12.221.3.1 Get() . . . 712
12.221.3.2 GetInteger() . . . 712
12.221.3.3 HasSection() . . . 712
12.221.3.4 HasValue() . . . 712
12.221.3.5 ParseError() . . . 712
12.222 InMemoryConfiguration Class Reference . . . 712
12.222.1 Detailed Description . . . 713
12.222.2 Member Function Documentation . . . 713
12.222.2.1 is_present() . . . 713
12.222.2.2 property() [1/10] . . . 713
12.222.2.3 property() [2/10] . . . 713
12.222.2.4 property() [3/10] . . . 713
12.222.2.5 property() [4/10] . . . 713
12.222.2.6 property() [5/10] . . . 714
12.222.2.7 property() [6/10] . . . 714
12.222.2.8 property() [7/10] . . . 714
12.222.2.9 property() [8/10] . . . 714
12.222.2.10 property() [9/10] . . . 714
12.222.2.11 property() [10/10] . . . 714
12.222.2.12 set_property() . . . 714
12.223 interleaved_byte_to_complex_byte Class Reference . . . 714
12.223.1 Detailed Description . . . 715
12.224 interleaved_byte_to_complex_short Class Reference . . . 715
12.224.1 Detailed Description . . . 715
12.225 interleaved_short_to_complex_short Class Reference . . . 715
12.225.1 Detailed Description . . . 716
12.226 IONGSMSChunkData Class Reference . . . 716
12.226.1 Detailed Description . . . 716
12.227 IONGSMSChunkUnpackingCtx< WT > Struct Template Reference . . . 717
12.227.1 Detailed Description . . . 717
12.227.2 Constructor & Destructor Documentation . . . 717
12.227.2.1 IONGSMSChunkUnpackingCtx() . . . 717
12.227.3 Member Function Documentation . . . 717
12.227.3.1 read_bits() . . . 717
12.227.3.2 shift_padding() . . . 717
12.227.4 Member Data Documentation . . . 717
12.227.4.1 bit_offset_ . . . 717
12.227.4.2 data_ . . . 718
12.227.4.3 read_lsb_first_ . . . 718
12.227.4.4 word_bitsize_ . . . 718

12.227.4.5 word_count_ . . . 718
12.228 IONGSMSFileSource Class Reference . . . 718
12.228.1 Detailed Description . . . 719
12.228.2 Member Typedef Documentation . . . 719
12.228.2.1 sptr . . . 719
12.229 IONGSMSSignalSource Class Reference . . . 719
12.229.1 Detailed Description . . . 720
12.229.2 Member Function Documentation . . . 720
12.229.2.1 connect() . . . 720
12.229.2.2 disconnect() . . . 720
12.229.2.3 get_left_block() . . . 720
12.229.2.4 get_right_block() [1/2] . . . 720
12.229.2.5 get_right_block() [2/2] . . . 720
12.229.2.6 getRfChannels() . . . 720
12.229.2.7 item_size() . . . 720
12.230 IshortToComplex Class Reference . . . 721
12.230.1 Detailed Description . . . 721
12.230.2 Member Function Documentation . . . 721
12.230.2.1 connect() . . . 721
12.230.2.2 disconnect() . . . 721
12.230.2.3 get_left_block() . . . 721
12.230.2.4 get_right_block() . . . 722
12.230.2.5 implementation() . . . 722
12.230.2.6 item_size() . . . 722
12.230.2.7 role() . . . 722
12.231 IshortToCshort Class Reference . . . 722
12.231.1 Detailed Description . . . 723
12.231.2 Member Function Documentation . . . 723
12.231.2.1 connect() . . . 723
12.231.2.2 disconnect() . . . 723
12.231.2.3 get_left_block() . . . 723
12.231.2.4 get_right_block() . . . 723
12.231.2.5 implementation() . . . 723
12.231.2.6 item_size() . . . 723
12.231.2.7 role() . . . 723
12.232 kernel_info_t Struct Reference . . . 723
12.232.1 Detailed Description . . . 724
12.232.2 Member Data Documentation . . . 724
12.232.2.1 dir . . . 724
12.232.2.2 in_place_possible . . . 724
12.232.2.3 kernel . . . 724
12.232.2.4 kernel_name . . . 724

12.232.2.5 lmem_size . . . 724
12.232.2.6 next . . . 724
12.232.2.7 num_workgroups . . . 724
12.232.2.8 num_workitems_per_workgroup . . . 724
12.232.2.9 num_xforms_per_workgroup . . . 724
12.233 Kf_Conf Class Reference . . . 725
12.233.1 Detailed Description . . . 725
12.233.2 Member Data Documentation . . . 726
12.233.2.1 bit_synchronization_time_limit_s . . . 726
12.233.2.2 carrier_disc_sd_rads . . . 726
12.233.2.3 carrier_freq_rate_sd_hz_s . . . 726
12.233.2.4 carrier_freq_sd_hz . . . 726
12.233.2.5 carrier_lock_test_smoother_alpha . . . 726
12.233.2.6 carrier_lock_test_smoother_samples . . . 726
12.233.2.7 carrier_lock_th . . . 726
12.233.2.8 carrier_phase_sd_rad . . . 726
12.233.2.9 cn0_min . . . 726
12.233.2.10 cn0_samples . . . 726
12.233.2.11 cn0_smoother_alpha . . . 726
12.233.2.12 cn0_smoother_samples . . . 726
12.233.2.13 code_disc_sd_chips . . . 727
12.233.2.14 code_phase_sd_chips . . . 727
12.233.2.15 dump . . . 727
12.233.2.16 dump_filename . . . 727
12.233.2.17 dump_mat . . . 727
12.233.2.18 early_late_space_chips . . . 727
12.233.2.19 early_late_space_narrow_chips . . . 727
12.233.2.20 enable_doppler_correction . . . 727
12.233.2.21 extend_correlation_symbols . . . 727
12.233.2.22 fs_in . . . 727
12.233.2.23 high_dyn . . . 727
12.233.2.24 init_carrier_freq_rate_sd_hz_s . . . 727
12.233.2.25 init_carrier_freq_sd_hz . . . 728
12.233.2.26 init_carrier_phase_sd_rad . . . 728
12.233.2.27 init_code_phase_sd_chips . . . 728
12.233.2.28 item_type . . . 728
12.233.2.29 max_carrier_lock_fail . . . 728
12.233.2.30 max_code_lock_fail . . . 728
12.233.2.31 pull_in_time_s . . . 728
12.233.2.32 signal . . . 728
12.233.2.33 slope . . . 728
12.233.2.34 smoother_length . . . 728

12.233.2.35 spc . . . 728
12.233.2.36 system . . . 728
12.233.2.37 track_pilot . . . 729
12.233.2.38 vector_length . . . 729
12.233.2.39 very_early_late_space_chips . . . 729
12.233.2.40 very_early_late_space_narrow_chips . . . 729
12.233.2.41 y_intercept . . . 729
12.234 kf_tracking Class Reference . . . 729
12.234.1 Detailed Description . . . 730
12.235 Kml_Printer Class Reference . . . 730
12.235.1 Detailed Description . . . 730
12.236 labsat23_source Class Reference . . . 730
12.236.1 Detailed Description . . . 730
12.237 LabsatSignalSource Class Reference . . . 731
12.237.1 Detailed Description . . . 731
12.237.2 Member Function Documentation . . . 732
12.237.2.1 connect() . . . 732
12.237.2.2 disconnect() . . . 732
12.237.2.3 get_left_block() . . . 732
12.237.2.4 get_right_block() [1/2] . . . 732
12.237.2.5 get_right_block() [2/2] . . . 732
12.237.2.6 getRfChannels() . . . 732
12.237.2.7 item_size() . . . 732
12.238 lex_t Struct Reference . . . 732
12.238.1 Detailed Description . . . 732
12.238.2 Member Data Documentation . . . 733
12.238.2.1 msgs . . . 733
12.238.2.2 n . . . 733
12.238.2.3 nmax . . . 733
12.239 lexeph_t Struct Reference . . . 733
12.239.1 Detailed Description . . . 733
12.239.2 Member Data Documentation . . . 733
12.239.2.1 acc . . . 733
12.239.2.2 af0 . . . 733
12.239.2.3 af1 . . . 733
12.239.2.4 health . . . 734
12.239.2.5 isc . . . 734
12.239.2.6 jerk . . . 734
12.239.2.7 pos . . . 734
12.239.2.8 sat . . . 734
12.239.2.9 tgd . . . 734
12.239.2.10 toe . . . 734

12.239.2.11 tof . . . 734
12.239.2.12 ura . . . 734
12.239.2.13 vel . . . 734
12.240 lexion_t Struct Reference . . . 734
12.240.1 Detailed Description . . . 735
12.240.2 Member Data Documentation . . . 735
12.240.2.1 coef . . . 735
12.240.2.2 pos0 . . . 735
12.240.2.3 t0 . . . 735
12.240.2.4 tspan . . . 735
12.241 lexmsg_t Struct Reference . . . 735
12.241.1 Detailed Description . . . 735
12.241.2 Member Data Documentation . . . 735
12.241.2.1 alert . . . 735
12.241.2.2 msg . . . 735
12.241.2.3 prn . . . 735
12.241.2.4 snr . . . 736
12.241.2.5 stat . . . 736
12.241.2.6 ttt . . . 736
12.241.2.7 type . . . 736
12.242 LimesdrSignalSource Class Reference . . . 736
12.242.1 Detailed Description . . . 737
12.242.2 Member Function Documentation . . . 737
12.242.2.1 connect() . . . 737
12.242.2.2 disconnect() . . . 737
12.242.2.3 get_left_block() . . . 737
12.242.2.4 get_right_block() . . . 737
12.242.2.5 item_size() . . . 737
12.243 MACK_header Class Reference . . . 738
12.243.1 Detailed Description . . . 738
12.243.2 Member Data Documentation . . . 738
12.243.2.1 cop . . . 738
12.243.2.2 macseq . . . 738
12.243.2.3 macseq_valid . . . 738
12.243.2.4 tag0 . . . 738
12.243.2.5 tag0_valid . . . 738
12.244 Mack_lookup Class Reference . . . 738
12.244.1 Detailed Description . . . 739
12.244.2 Constructor & Destructor Documentation . . . 739
12.244.2.1 Mack_lookup() . . . 739
12.244.3 Member Data Documentation . . . 739
12.244.3.1 msg . . . 739

12.244.3.2 nt . . . 739
12.244.3.3 sequence1 . . . 739
12.244.3.4 sequence2 . . . 739
12.245 MACK_message Class Reference . . . 739
12.245.1 Detailed Description . . . 739
12.245.2 Member Data Documentation . . . 740
12.245.2.1 allowed_adkds . . . 740
12.245.2.2 header . . . 740
12.245.2.3 key . . . 740
12.245.2.4 nmas . . . 740
12.245.2.5 PRNa . . . 740
12.245.2.6 tag_and_info . . . 740
12.245.2.7 TOW . . . 740
12.245.2.8 WN . . . 740
12.246 MACK_tag_and_info Class Reference . . . 740
12.246.1 Detailed Description . . . 740
12.246.2 Member Data Documentation . . . 741
12.246.2.1 counter . . . 741
12.246.2.2 tag . . . 741
12.246.2.3 tag_info . . . 741
12.246.2.4 valid . . . 741
12.247 MACK_tag_info Class Reference . . . 741
12.247.1 Detailed Description . . . 741
12.247.2 Member Data Documentation . . . 741
12.247.2.1 ADKD . . . 741
12.247.2.2 cop . . . 741
12.247.2.3 PRN_d . . . 741
12.248 matlab_type_traits< T > Struct Template Reference . . . 742
12.248.1 Detailed Description . . . 742
12.249 matlab_type_traits< double > Struct Reference . . . 742
12.249.1 Detailed Description . . . 742
12.249.2 Member Data Documentation . . . 742
12.249.2.1 class_type . . . 742
12.249.2.2 data_type . . . 742
12.250 matlab_type_traits< float > Struct Reference . . . 742
12.250.1 Detailed Description . . . 742
12.250.2 Member Data Documentation . . . 742
12.250.2.1 class_type . . . 742
12.250.2.2 data_type . . . 743
12.251 matlab_type_traits< int32_t > Struct Reference . . . 743
12.251.1 Detailed Description . . . 743
12.251.2 Member Data Documentation . . . 743

12.251.2.1 class_type . . . 743
12.251.2.2 data_type . . . 743
12.252 matlab_type_traits< int64_t > Struct Reference . . . 743
12.252.1 Detailed Description . . . 743
12.252.2 Member Data Documentation . . . 743
12.252.2.1 class_type . . . 743
12.252.2.2 data_type . . . 743
12.253 matlab_type_traits< uint32_t > Struct Reference . . . 744
12.253.1 Detailed Description . . . 744
12.253.2 Member Data Documentation . . . 744
12.253.2.1 class_type . . . 744
12.253.2.2 data_type . . . 744
12.254 matlab_type_traits< uint64_t > Struct Reference . . . 744
12.254.1 Detailed Description . . . 744
12.254.2 Member Data Documentation . . . 744
12.254.2.1 class_type . . . 744
12.254.2.2 data_type . . . 744
12.255 matlab_type_traits< uint8_t > Struct Reference . . . 744
12.255.1 Detailed Description . . . 745
12.255.2 Member Data Documentation . . . 745
12.255.2.1 class_type . . . 745
12.255.2.2 data_type . . . 745
12.256 MAX2771EVKITSignalSourceFPGA Class Reference . . . 745
12.256.1 Detailed Description . . . 746
12.256.2 Member Function Documentation . . . 746
12.256.2.1 connect() . . . 746
12.256.2.2 disconnect() . . . 746
12.256.2.3 get_left_block() . . . 746
12.256.2.4 get_right_block() . . . 746
12.256.2.5 item_size() . . . 746
12.257 MmseResamplerConditioner Class Reference . . . 746
12.257.1 Detailed Description . . . 747
12.257.2 Member Function Documentation . . . 747
12.257.2.1 connect() . . . 747
12.257.2.2 disconnect() . . . 747
12.257.2.3 get_left_block() . . . 747
12.257.2.4 get_right_block() . . . 747
12.257.2.5 implementation() . . . 747
12.257.2.6 item_size() . . . 747
12.257.2.7 role() . . . 748
12.258 ModelFunction Class Reference . . . 748
12.258.1 Detailed Description . . . 748

12.258.2 Constructor & Destructor Documentation . . . 748
12.258.2.1 ModelFunction() . . . 748
12.259 Monitor_Ephemeris_Udp_Sink Class Reference . . . 748
12.259.1 Detailed Description . . . 748
12.260 Monitor_Pvt Class Reference . . . 748
12.260.1 Detailed Description . . . 749
12.260.2 Member Function Documentation . . . 750
12.260.2.1 serialize() . . . 750
12.260.3 Member Data Documentation . . . 750
12.260.3.1 AR_ratio_factor . . . 750
12.260.3.2 AR_ratio_threshold . . . 750
12.260.3.3 cog . . . 750
12.260.3.4 cov_xx . . . 750
12.260.3.5 cov_xy . . . 750
12.260.3.6 cov_yy . . . 750
12.260.3.7 cov_yz . . . 750
12.260.3.8 cov_zx . . . 750
12.260.3.9 cov_zz . . . 750
12.260.3.10 galhas_status . . . 751
12.260.3.11 gdop . . . 751
12.260.3.12 geohash . . . 751
12.260.3.13 hdop . . . 751
12.260.3.14 height . . . 751
12.260.3.15 latitude . . . 751
12.260.3.16 longitude . . . 751
12.260.3.17 pdop . . . 751
12.260.3.18 pos_x . . . 751
12.260.3.19 pos_y . . . 751
12.260.3.20 pos_z . . . 751
12.260.3.21 RX_time . . . 751
12.260.3.22 solution_status . . . 752
12.260.3.23 solution_type . . . 752
12.260.3.24 TOW_at_current_symbol_ms . . . 752
12.260.3.25 tracked_satellites . . . 752
12.260.3.26 user_clk_drift_ppm . . . 752
12.260.3.27 user_clk_offset . . . 752
12.260.3.28 utc_time . . . 752
12.260.3.29 valid_sats . . . 752
12.260.3.30 vdop . . . 752
12.260.3.31 vel_e . . . 752
12.260.3.32 vel_n . . . 752
12.260.3.33 vel_u . . . 752

12.260.3.34 vel_x . . . 753
12.260.3.35 vel_y . . . 753
12.260.3.36 vel_z . . . 753
12.260.3.37 week . . . 753
12.261 Monitor_Pvt_Udp_Sink Class Reference . . . 753
12.261.1 Detailed Description . . . 753
12.262 msm_h_t Struct Reference . . . 753
12.262.1 Detailed Description . . . 753
12.262.2 Member Data Documentation . . . 754
12.262.2.1 cellmask . . . 754
12.262.2.2 clk_ext . . . 754
12.262.2.3 clk_str . . . 754
12.262.2.4 iod . . . 754
12.262.2.5 nsat . . . 754
12.262.2.6 nsig . . . 754
12.262.2.7 sats . . . 754
12.262.2.8 sigs . . . 754
12.262.2.9 smooth . . . 754
12.262.2.10 time_s . . . 754
12.262.2.11 tint_s . . . 754
12.263 mt1_header Struct Reference . . . 755
12.263.1 Detailed Description . . . 755
12.263.2 Member Data Documentation . . . 755
12.263.2.1 clock_fullset_flag . . . 755
12.263.2.2 clock_subset_flag . . . 755
12.263.2.3 code_bias_flag . . . 755
12.263.2.4 iod_set_id . . . 755
12.263.2.5 mask_flag . . . 755
12.263.2.6 mask_id . . . 755
12.263.2.7 orbit_correction_flag . . . 755
12.263.2.8 phase_bias_flag . . . 755
12.263.2.9 reserved . . . 756
12.263.2.10 toh . . . 756
12.264 MultichannelFileSignalSource Class Reference . . . 756
12.264.1 Detailed Description . . . 757
12.264.2 Member Function Documentation . . . 757
12.264.2.1 connect() . . . 757
12.264.2.2 disconnect() . . . 757
12.264.2.3 filename() . . . 757
12.264.2.4 get_left_block() . . . 757
12.264.2.5 get_right_block() . . . 757
12.264.2.6 item_size() . . . 757

12.264.2.7 item_type() . . . 757
12.264.2.8 repeat() . . . 757
12.264.2.9 samples() . . . 758
12.264.2.10 sampling_frequency() . . . 758
12.265 nav_message_monitor Class Reference . . . 758
12.265.1 Detailed Description . . . 758
12.265.2 Constructor & Destructor Documentation . . . 758
12.265.2.1 ~nav_message_monitor() . . . 758
12.266 Nav_Message_Packet Class Reference . . . 758
12.266.1 Detailed Description . . . 759
12.266.2 Constructor & Destructor Documentation . . . 759
12.266.2.1 Nav_Message_Packet() [1/3] . . . 759
12.266.2.2 ~Nav_Message_Packet() . . . 759
12.266.2.3 Nav_Message_Packet() [2/3] . . . 759
12.266.2.4 Nav_Message_Packet() [3/3] . . . 759
12.266.3 Member Function Documentation . . . 760
12.266.3.1 operator=() [1/2] . . . 760
12.266.3.2 operator=() [2/2] . . . 760
12.266.4 Member Data Documentation . . . 760
12.266.4.1 nav_message . . . 760
12.266.4.2 prn . . . 760
12.266.4.3 signal . . . 760
12.266.4.4 system . . . 760
12.266.4.5 tow_at_current_symbol_ms . . . 760
12.267 Nav_Message_Udp_Sink Class Reference . . . 761
12.267.1 Detailed Description . . . 761
12.268 nav_t Struct Reference . . . 761
12.268.1 Detailed Description . . . 762
12.268.2 Member Data Documentation . . . 762
12.268.2.1 alm . . . 762
12.268.2.2 cbias . . . 762
12.268.2.3 dgps . . . 762
12.268.2.4 eph . . . 762
12.268.2.5 erp . . . 762
12.268.2.6 fcb . . . 762
12.268.2.7 geph . . . 762
12.268.2.8 glo_cpbias . . . 763
12.268.2.9 glo_fcn . . . 763
12.268.2.10 ion_bdgim . . . 763
12.268.2.11 ion_bdgim_valid . . . 763
12.268.2.12 ion_cmp . . . 763
12.268.2.13 ion_gal . . . 763

12.268.2.14 ion_gps . . . 763
12.268.2.15 ion_irn . . . 763
12.268.2.16 ion_qzs . . . 763
12.268.2.17 lam . . . 763
12.268.2.18 leaps . . . 763
12.268.2.19 lexeph . . . 763
12.268.2.20 lexion . . . 764
12.268.2.21 n . . . 764
12.268.2.22 na . . . 764
12.268.2.23 namax . . . 764
12.268.2.24 nc . . . 764
12.268.2.25 ncmax . . . 764
12.268.2.26 ne . . . 764
12.268.2.27 nemax . . . 764
12.268.2.28 nf . . . 764
12.268.2.29 nfmax . . . 764
12.268.2.30 ng . . . 764
12.268.2.31 ngmax . . . 764
12.268.2.32 nmax . . . 765
12.268.2.33 ns . . . 765
12.268.2.34 nsmax . . . 765
12.268.2.35 nt . . . 765
12.268.2.36 ntmax . . . 765
12.268.2.37 pclk . . . 765
12.268.2.38 pcvs . . . 765
12.268.2.39 peph . . . 765
12.268.2.40 pppcorr . . . 765
12.268.2.41 rbias . . . 765
12.268.2.42 sbsion . . . 765
12.268.2.43 sbssat . . . 765
12.268.2.44 seph . . . 766
12.268.2.45 ssr . . . 766
12.268.2.46 tec . . . 766
12.268.2.47 utc_cmp . . . 766
12.268.2.48 utc_gal . . . 766
12.268.2.49 utc_glo . . . 766
12.268.2.50 utc_gps . . . 766
12.268.2.51 utc_irn . . . 766
12.268.2.52 utc_qzs . . . 766
12.268.2.53 utc_sbs . . . 766
12.268.2.54 wlbias . . . 766
12.269 Nmea_Printer Class Reference . . . 767

12.269.1 Detailed Description . . . 767
12.269.2 Constructor & Destructor Documentation . . . 767
12.269.2.1 Nmea_Printer() . . . 767
12.269.2.2 ~Nmea_Printer() . . . 767
12.269.3 Member Function Documentation . . . 767
12.269.3.1 get_GPGGA() . . . 767
12.269.3.2 get_GPGSA() . . . 768
12.269.3.3 Print_Nmea_Line() . . . 768
12.270 Notch Class Reference . . . 768
12.270.1 Detailed Description . . . 768
12.271 NotchFilter Class Reference . . . 768
12.271.1 Detailed Description . . . 769
12.271.2 Member Function Documentation . . . 769
12.271.2.1 connect() . . . 769
12.271.2.2 disconnect() . . . 769
12.271.2.3 get_left_block() . . . 769
12.271.2.4 get_right_block() . . . 769
12.271.2.5 implementation() . . . 769
12.271.2.6 item_size() . . . 769
12.271.2.7 role() . . . 770
12.272 NotchFilterLite Class Reference . . . 770
12.272.1 Detailed Description . . . 770
12.272.2 Member Function Documentation . . . 770
12.272.2.1 connect() . . . 770
12.272.2.2 disconnect() . . . 770
12.272.2.3 get_left_block() . . . 771
12.272.2.4 get_right_block() . . . 771
12.272.2.5 implementation() . . . 771
12.272.2.6 item_size() . . . 771
12.272.2.7 role() . . . 771
12.273 NotchLite Class Reference . . . 771
12.273.1 Detailed Description . . . 771
12.274 NsrFileSignalSource Class Reference . . . 772
12.274.1 Detailed Description . . . 774
12.274.2 Member Function Documentation . . . 774
12.274.2.1 create_file_source_hook() . . . 774
12.274.2.2 itemTypeToSize() . . . 774
12.274.2.3 packetsPerSample() . . . 774
12.274.2.4 pre_connect_hook() . . . 774
12.274.2.5 pre_disconnect_hook() . . . 774
12.274.2.6 source() . . . 774
12.275 NTLabFileSignalSource Class Reference . . . 774

12.275.1 Detailed Description . . . 776
12.275.2 Member Function Documentation . . . 777
12.275.2.1 create_file_source_hook() . . . 777
12.275.2.2 get_left_block() . . . 777
12.275.2.3 get_right_block() . . . 777
12.275.2.4 itemTypeToSize() . . . 777
12.275.2.5 packetsPerSample() . . . 777
12.275.2.6 pre_connect_hook() . . . 777
12.275.2.7 pre_disconnect_hook() . . . 777
12.275.2.8 source() . . . 777
12.276 Ntrip_Rtcm_Client Class Reference . . . 777
12.276.1 Detailed Description . . . 778
12.276.2 Friends And Related Symbol Documentation . . . 778
12.276.2.1 Ntrip_Rtcm_Client_Test_Access . . . 778
12.277 Ntrip_Rtcm_Client_Config Struct Reference . . . 778
12.277.1 Detailed Description . . . 779
12.277.2 Member Function Documentation . . . 779
12.277.2.1 validate() . . . 779
12.277.3 Member Data Documentation . . . 779
12.277.3.1 enabled . . . 779
12.277.3.2 gga_period_ms . . . 779
12.277.3.3 host . . . 779
12.277.3.4 max_age_s . . . 779
12.277.3.5 mountpoint . . . 779
12.277.3.6 password . . . 779
12.277.3.7 port . . . 779
12.277.3.8 reconnect_interval_ms . . . 780
12.277.3.9 send_gga . . . 780
12.277.3.10 station_id . . . 780
12.277.3.11 timeout_ms . . . 780
12.277.3.12 tls_enabled . . . 780
12.277.3.13 username . . . 780
12.277.3.14 version . . . 780
12.278 Ntrip_Rtcm_Client_Status Struct Reference . . . 780
12.278.1 Detailed Description . . . 780
12.278.2 Member Data Documentation . . . 781
12.278.2.1 bytes_received . . . 781
12.278.2.2 connected . . . 781
12.278.2.3 decoder_errors . . . 781
12.278.2.4 message . . . 781
12.278.2.5 observation_epochs_decoded . . . 781
12.278.2.6 reconnect_count . . . 781

12.278.2.7 rtcm_messages_decoded . . . 781
12.278.2.8 running . . . 781
12.278.2.9 state . . . 781
12.279 Ntrip_Rtcm_Snapshot Struct Reference . . . 781
12.279.1 Detailed Description . . . 782
12.279.2 Member Function Documentation . . . 782
12.279.2.1 refresh_age() . . . 782
12.279.3 Member Data Documentation . . . 782
12.279.3.1 age_s . . . 782
12.279.3.2 antenna_height_m . . . 782
12.279.3.3 base_position_ecef_m . . . 782
12.279.3.4 base_position_message_type . . . 782
12.279.3.5 ephemerides . . . 782
12.279.3.6 fresh . . . 783
12.279.3.7 generation . . . 783
12.279.3.8 has_base_position . . . 783
12.279.3.9 has_observations . . . 783
12.279.3.10 itrf . . . 783
12.279.3.11 observation_time . . . 783
12.279.3.12 observations . . . 783
12.279.3.13 station_id . . . 783
12.280 Ntrip_Rtk_Signal Struct Reference . . . 783
12.280.1 Detailed Description . . . 784
12.280.2 Member Data Documentation . . . 784
12.280.2.1 band_slot . . . 784
12.280.2.2 flag . . . 784
12.280.2.3 signal . . . 784
12.280.2.4 system . . . 784
12.281 ntrip_t Struct Reference . . . 784
12.281.1 Detailed Description . . . 785
12.281.2 Member Data Documentation . . . 785
12.281.2.1 buff . . . 785
12.281.2.2 chunk_remaining . . . 785
12.281.2.3 chunk_state . . . 785
12.281.2.4 chunked . . . 785
12.281.2.5 host . . . 785
12.281.2.6 mntpnt . . . 785
12.281.2.7 nb . . . 785
12.281.2.8 passwd . . . 785
12.281.2.9 port . . . 786
12.281.2.10 request_buff . . . 786
12.281.2.11 request_length . . . 786

12.281.2.12 request_offset . . . 786
12.281.2.13 request_sent . . . 786
12.281.2.14 response_received . . . 786
12.281.2.15 state . . . 786
12.281.2.16 str . . . 786
12.281.2.17 tcp . . . 786
12.281.2.18 tls_ctx . . . 786
12.281.2.19 tls_enabled . . . 786
12.281.2.20 type . . . 786
12.281.2.21 url . . . 787
12.281.2.22 user . . . 787
12.281.2.23 v1_retry_requested . . . 787
12.281.2.24 version . . . 787
12.282 Obs_Conf Class Reference . . . 787
12.282.1 Detailed Description . . . 787
12.282.2 Member Data Documentation . . . 787
12.282.2.1 always_output_gs . . . 787
12.282.2.2 dump . . . 787
12.282.2.3 dump_filename . . . 787
12.282.2.4 dump_mat . . . 788
12.282.2.5 enable_carrier_smoothing . . . 788
12.282.2.6 enable_E6 . . . 788
12.282.2.7 enable_monitor . . . 788
12.282.2.8 nchannels_in . . . 788
12.282.2.9 nchannels_out . . . 788
12.282.2.10 observable_interval_ms . . . 788
12.282.2.11 smoothing_factor . . . 788
12.283 obs_t Struct Reference . . . 788
12.283.1 Detailed Description . . . 788
12.283.2 Member Data Documentation . . . 788
12.283.2.1 data . . . 788
12.283.2.2 n . . . 789
12.283.2.3 nmax . . . 789
12.284 obsd_t Struct Reference . . . 789
12.284.1 Detailed Description . . . 789
12.284.2 Member Data Documentation . . . 789
12.284.2.1 code . . . 789
12.284.2.2 D . . . 789
12.284.2.3 L . . . 789
12.284.2.4 LLI . . . 789
12.284.2.5 Lstd . . . 789
12.284.2.6 P . . . 790

12.284.2.7 Pstd . . . 790
12.284.2.8 rcv . . . 790
12.284.2.9 sat . . . 790
12.284.2.10 SNR . . . 790
12.284.2.11 time . . . 790
12.285 ObservablesInterface Class Reference . . . 790
12.285.1 Detailed Description . . . 791
12.286 opt_t Struct Reference . . . 791
12.286.1 Detailed Description . . . 791
12.286.2 Member Data Documentation . . . 791
12.286.2.1 comment . . . 791
12.286.2.2 format . . . 791
12.286.2.3 name . . . 791
12.286.2.4 var . . . 791
12.287 OsmosdrSignalSource Class Reference . . . 792
12.287.1 Detailed Description . . . 792
12.287.2 Member Function Documentation . . . 793
12.287.2.1 connect() . . . 793
12.287.2.2 disconnect() . . . 793
12.287.2.3 get_left_block() . . . 793
12.287.2.4 get_right_block() . . . 793
12.287.2.5 item_size() . . . 793
12.288 Osnma_Crypto_Material_Manager Class Reference . . . 793
12.288.1 Detailed Description . . . 794
12.289 OSNMA_data Class Reference . . . 794
12.289.1 Detailed Description . . . 794
12.289.2 Member Data Documentation . . . 794
12.289.2.1 d_dsm_header . . . 794
12.289.2.2 d_dsm_kroot_message . . . 794
12.289.2.3 d_dsm_kroot_new_message . . . 794
12.289.2.4 d_dsm_pkr_message . . . 794
12.289.2.5 d_mack_message . . . 795
12.289.2.6 d_nav_data . . . 795
12.289.2.7 d_nma_header . . . 795
12.290 OSNMA_DSM_Reader Class Reference . . . 795
12.290.1 Detailed Description . . . 795
12.291 Osnma_Helper Class Reference . . . 796
12.291.1 Detailed Description . . . 796
12.291.2 Member Data Documentation . . . 796
12.291.2.1 GST_START_EPOCH . . . 796
12.292 Osnma_Merkle_Tree_Material Struct Reference . . . 796
12.292.1 Detailed Description . . . 796

12.292.2 Member Data Documentation . . . 797
12.292.2.1 applicability . . . 797
12.292.2.2 hash_function . . . 797
12.292.2.3 public_keys . . . 797
12.292.2.4 root . . . 797
12.292.2.5 source . . . 797
12.292.2.6 state . . . 797
12.292.2.7 uid . . . 797
12.292.2.8 valid . . . 797
12.292.2.9 xml_path . . . 797
12.293 OSNMA_msg Class Reference . . . 797
12.293.1 Detailed Description . . . 798
12.293.2 Member Data Documentation . . . 798
12.293.2.1 hkroot . . . 798
12.293.2.2 mack . . . 798
12.293.2.3 page_valid . . . 798
12.293.2.4 page_validity_available . . . 798
12.293.2.5 PRN . . . 798
12.293.2.6 TOW_sf0 . . . 798
12.293.2.7 WN_sf0 . . . 798
12.294 osnma_msg_receiver Class Reference . . . 798
12.294.1 Detailed Description . . . 799
12.294.2 Constructor & Destructor Documentation . . . 799
12.294.2.1 ∼osnma_msg_receiver() . . . 799
12.294.3 Member Function Documentation . . . 799
12.294.3.1 msg_handler_osnma() . . . 799
12.294.3.2 read_merkle_xml() . . . 799
12.294.3.3 set_merkle_root() . . . 800
12.294.3.4 verify_dsm_pkr() . . . 800
12.294.4 Friends And Related Symbol Documentation . . . 800
12.294.4.1 OsnmaTestVectors . . . 800
12.295 OSNMA_NavData Class Reference . . . 800
12.295.1 Detailed Description . . . 801
12.295.2 Constructor & Destructor Documentation . . . 801
12.295.2.1 OSNMA_NavData() . . . 801
12.295.3 Member Function Documentation . . . 801
12.295.3.1 clear_tag_accumulation_metadata() . . . 801
12.295.3.2 continue_tag_accumulation() . . . 801
12.295.3.3 get_accumulated_tag_adkd() . . . 801
12.295.3.4 get_first_accumulated_tag_gst() . . . 801
12.295.3.5 get_IOD_nav() . . . 801
12.295.3.6 get_last_accumulated_tag_gst() . . . 801

12.295.3.7 get_last_received_TOW() . . . 801
12.295.3.8 get_last_received_WN() . . . 801
12.295.3.9 get_prn_d() . . . 802
12.295.3.10 get_tow_sf0() . . . 802
12.295.3.11 get_verified_bits() . . . 802
12.295.3.12 get_verified_status() . . . 802
12.295.3.13 get_wn_sf0() . . . 802
12.295.3.14 has_tag_accumulation() . . . 802
12.295.3.15 reset_tag_accumulation() . . . 802
12.295.3.16 set_ephemeris_data() . . . 802
12.295.3.17 set_IOD_nav() . . . 802
12.295.3.18 set_last_received_TOW() . . . 802
12.295.3.19 set_last_received_WN() . . . 802
12.295.3.20 set_prn_d() . . . 803
12.295.3.21 set_tow_sf0() . . . 803
12.295.3.22 set_update_verified_bits() . . . 803
12.295.3.23 set_utc_data() . . . 803
12.295.3.24 set_verified_status() . . . 803
12.295.3.25 set_wn_sf0() . . . 803
12.295.3.26 start_tag_accumulation() . . . 803
12.295.4 Member Data Documentation . . . 803
12.295.4.1 nav_data_id . . . 803
12.296 OSNMA_NavDataManager Class Reference . . . 803
12.296.1 Detailed Description . . . 804
12.297 Osnma_Public_Key_Material Struct Reference . . . 804
12.297.1 Detailed Description . . . 804
12.297.2 Member Data Documentation . . . 804
12.297.2.1 applicability . . . 804
12.297.2.2 compressed_key . . . 804
12.297.2.3 fingerprint_sha256 . . . 804
12.297.2.4 key_type . . . 805
12.297.2.5 npkt . . . 805
12.297.2.6 npkt_valid . . . 805
12.297.2.7 pem_path . . . 805
12.297.2.8 pkid . . . 805
12.297.2.9 pkid_valid . . . 805
12.297.2.10 product_uid . . . 805
12.297.2.11 source . . . 805
12.297.2.12 state . . . 805
12.297.2.13 valid . . . 805
12.298 Pass_Through Class Reference . . . 805
12.298.1 Detailed Description . . . 806

12.298.2 Member Function Documentation . . . 806
12.298.2.1 connect() . . . 806
12.298.2.2 disconnect() . . . 806
12.298.2.3 get_left_block() . . . 806
12.298.2.4 get_right_block() . . . 806
12.298.2.5 implementation() . . . 807
12.298.2.6 item_size() . . . 807
12.298.2.7 item_type() . . . 807
12.298.2.8 role() . . . 807
12.299 pclk_t Struct Reference . . . 807
12.299.1 Detailed Description . . . 807
12.299.2 Member Data Documentation . . . 807
12.299.2.1 clk . . . 807
12.299.2.2 index . . . 807
12.299.2.3 std . . . 807
12.299.2.4 time . . . 808
12.300 pcps_acquisition Class Reference . . . 808
12.300.1 Detailed Description . . . 809
12.300.2 Member Function Documentation . . . 809
12.300.2.1 cuda_grid_count() . . . 809
12.300.2.2 cuda_ready() . . . 809
12.300.2.3 general_work() . . . 809
12.300.2.4 mag() . . . 809
12.300.2.5 set_active() . . . 809
12.300.2.6 set_channel() . . . 809
12.300.2.7 set_channel_fsm() . . . 810
12.300.2.8 set_doppler_center() . . . 810
12.300.2.9 set_doppler_num_bins() . . . 810
12.300.2.10 set_gnss_synchro() . . . 810
12.300.2.11 set_local_code() . . . 810
12.301 pcps_acquisition_fine_doppler_cc Class Reference . . . 811
12.301.1 Detailed Description . . . 812
12.301.2 Constructor & Destructor Documentation . . . 812
12.301.2.1 ~pcps_acquisition_fine_doppler_cc() . . . 812
12.301.3 Member Function Documentation . . . 812
12.301.3.1 general_work() . . . 812
12.301.3.2 mag() . . . 812
12.301.3.3 set_active() . . . 812
12.301.3.4 set_channel() . . . 812
12.301.3.5 set_channel_fsm() . . . 813
12.301.3.6 set_gnss_synchro() . . . 813
12.301.3.7 set_local_code() . . . 813

12.302 pcps_acquisition_fpga Class Reference . . . 813
12.302.1 Detailed Description . . . 814
12.302.2 Constructor & Destructor Documentation . . . 814
12.302.2.1 ~pcps_acquisition_fpga() . . . 814
12.302.3 Member Function Documentation . . . 814
12.302.3.1 init() . . . 814
12.302.3.2 mag() . . . 814
12.302.3.3 reset_acquisition() . . . 814
12.302.3.4 set_active() . . . 814
12.302.3.5 set_channel() . . . 815
12.302.3.6 set_channel_fsm() . . . 815
12.302.3.7 set_doppler_center() . . . 815
12.302.3.8 set_gnss_synchro() . . . 815
12.302.3.9 set_local_code() . . . 815
12.302.3.10 stop_acquisition() . . . 815
12.303 pcps_assisted_acquisition_cc Class Reference . . . 816
12.303.1 Detailed Description . . . 816
12.303.2 Constructor & Destructor Documentation . . . 817
12.303.2.1 ~pcps_assisted_acquisition_cc() . . . 817
12.303.3 Member Function Documentation . . . 817
12.303.3.1 general_work() . . . 817
12.303.3.2 mag() . . . 817
12.303.3.3 set_active() . . . 817
12.303.3.4 set_channel() . . . 817
12.303.3.5 set_channel_fsm() . . . 817
12.303.3.6 set_gnss_synchro() . . . 818
12.303.3.7 set_local_code() . . . 818
12.304 pcps_cccwsr_acquisition_cc Class Reference . . . 818
12.304.1 Detailed Description . . . 819
12.304.2 Constructor & Destructor Documentation . . . 819
12.304.2.1 ~pcps_cccwsr_acquisition_cc() . . . 819
12.304.3 Member Function Documentation . . . 819
12.304.3.1 general_work() . . . 819
12.304.3.2 mag() . . . 819
12.304.3.3 set_active() . . . 819
12.304.3.4 set_channel() . . . 820
12.304.3.5 set_channel_fsm() . . . 820
12.304.3.6 set_gnss_synchro() . . . 820
12.304.3.7 set_local_code() . . . 820
12.305 pcps_opencl_acquisition_cc Class Reference . . . 821
12.305.1 Detailed Description . . . 821
12.305.2 Constructor & Destructor Documentation . . . 822

12.305.2.1 ~pcps_opencl_acquisition_cc() . . . 822
12.305.3 Member Function Documentation . . . 822
12.305.3.1 general_work() . . . 822
12.305.3.2 mag() . . . 822
12.305.3.3 opencl_ready() . . . 822
12.305.3.4 set_active() . . . 822
12.305.3.5 set_channel() . . . 822
12.305.3.6 set_channel_fsm() . . . 822
12.305.3.7 set_gnss_synchro() . . . 823
12.305.3.8 set_local_code() . . . 823
12.306 pcps_quicksync_acquisition_cc Class Reference . . . 823
12.306.1 Detailed Description . . . 824
12.306.2 Constructor & Destructor Documentation . . . 824
12.306.2.1 ~pcps_quicksync_acquisition_cc() . . . 824
12.306.3 Member Function Documentation . . . 824
12.306.3.1 general_work() . . . 824
12.306.3.2 mag() . . . 824
12.306.3.3 set_active() . . . 825
12.306.3.4 set_channel() . . . 825
12.306.3.5 set_channel_fsm() . . . 825
12.306.3.6 set_gnss_synchro() . . . 825
12.306.3.7 set_local_code() . . . 825
12.307 pcps_tong_acquisition_cc Class Reference . . . 826
12.307.1 Detailed Description . . . 826
12.307.2 Constructor & Destructor Documentation . . . 827
12.307.2.1 ~pcps_tong_acquisition_cc() . . . 827
12.307.3 Member Function Documentation . . . 827
12.307.3.1 general_work() . . . 827
12.307.3.2 mag() . . . 827
12.307.3.3 set_active() . . . 827
12.307.3.4 set_channel() . . . 827
12.307.3.5 set_channel_fsm() . . . 827
12.307.3.6 set_gnss_synchro() . . . 828
12.307.3.7 set_local_code() . . . 828
12.308 PcpsAcquisitionAdapter Class Reference . . . 828
12.308.1 Detailed Description . . . 829
12.308.2 Member Function Documentation . . . 829
12.308.2.1 connect() . . . 829
12.308.2.2 disconnect() . . . 829
12.308.2.3 get_left_block() . . . 829
12.308.2.4 get_right_block() . . . 829
12.308.2.5 implementation() . . . 830

12.308.2.6 item_size() . . . 830
12.308.2.7 mag() . . . 830
12.308.2.8 reset() . . . 830
12.308.2.9 role() . . . 830
12.308.2.10 set_channel() . . . 830
12.308.2.11 set_channel_fsm() . . . 830
12.308.2.12 set_doppler_center() . . . 830
12.308.2.13 set_doppler_num_bins() . . . 830
12.308.2.14 set_gnss_synchro() . . . 831
12.308.2.15 set_local_code() . . . 831
12.308.2.16 set_resampler_latency() . . . 831
12.308.2.17 stop_acquisition() . . . 831
12.309 PcpsAcquisitionAdapterCustom Class Reference . . . 831
12.309.1 Detailed Description . . . 832
12.309.2 Member Function Documentation . . . 832
12.309.2.1 connect() . . . 832
12.309.2.2 disconnect() . . . 832
12.309.2.3 get_left_block() . . . 832
12.309.2.4 get_right_block() . . . 833
12.309.2.5 implementation() . . . 833
12.309.2.6 item_size() . . . 833
12.309.2.7 mag() . . . 833
12.309.2.8 reset() . . . 833
12.309.2.9 role() . . . 833
12.309.2.10 set_channel() . . . 833
12.309.2.11 set_channel_fsm() . . . 833
12.309.2.12 set_gnss_synchro() . . . 833
12.309.2.13 set_local_code() . . . 834
12.309.2.14 set_resampler_latency() . . . 834
12.309.2.15 stop_acquisition() . . . 834
12.310 PcpsAcquisitionAdapterFpga Class Reference . . . 834
12.310.1 Detailed Description . . . 835
12.310.2 Member Function Documentation . . . 835
12.310.2.1 connect() . . . 835
12.310.2.2 disconnect() . . . 835
12.310.2.3 get_left_block() . . . 835
12.310.2.4 get_right_block() . . . 835
12.310.2.5 implementation() . . . 835
12.310.2.6 item_size() . . . 835
12.310.2.7 mag() . . . 836
12.310.2.8 reset() . . . 836
12.310.2.9 role() . . . 836

12.310.2.10 set_channel() . . . 836
12.310.2.11 set_channel_fsm() . . . 836
12.310.2.12 set_doppler_center() . . . 836
12.310.2.13 set_gnss_synchro() . . . 836
12.310.2.14 set_local_code() . . . 836
12.310.2.15 set_resampler_latency() . . . 836
12.310.2.16 stop_acquisition() . . . 836
12.311 pcv_t Struct Reference . . . 837
12.311.1 Detailed Description . . . 837
12.311.2 Member Data Documentation . . . 837
12.311.2.1 code . . . 837
12.311.2.2 off . . . 837
12.311.2.3 sat . . . 837
12.311.2.4 te . . . 837
12.311.2.5 ts . . . 837
12.311.2.6 type . . . 837
12.311.2.7 var . . . 837
12.312 pcvs_t Struct Reference . . . 838
12.312.1 Detailed Description . . . 838
12.312.2 Member Data Documentation . . . 838
12.312.2.1 n . . . 838
12.312.2.2 nmax . . . 838
12.312.2.3 pcv . . . 838
12.313 peph_t Struct Reference . . . 838
12.313.1 Detailed Description . . . 838
12.313.2 Member Data Documentation . . . 838
12.313.2.1 cov . . . 838
12.313.2.2 index . . . 838
12.313.2.3 pos . . . 839
12.313.2.4 std . . . 839
12.313.2.5 time . . . 839
12.313.2.6 vco . . . 839
12.313.2.7 vel . . . 839
12.313.2.8 vst . . . 839
12.314 PlutosdrSignalSource Class Reference . . . 839
12.314.1 Detailed Description . . . 840
12.314.2 Member Function Documentation . . . 840
12.314.2.1 connect() . . . 840
12.314.2.2 disconnect() . . . 840
12.314.2.3 get_left_block() . . . 840
12.314.2.4 get_right_block() . . . 840
12.314.2.5 item_size() . . . 840

12.315 PocketSdrSignalSource Class Reference . . . 841
12.315.1 Detailed Description . . . 841
12.315.2 Member Function Documentation . . . 842
12.315.2.1 connect() . . . 842
12.315.2.2 disconnect() . . . 842
12.315.2.3 get_left_block() . . . 842
12.315.2.4 get_right_block() . . . 842
12.315.2.5 item_size() . . . 842
12.316 pppcorr_t Struct Reference . . . 842
12.316.1 Detailed Description . . . 842
12.316.2 Member Data Documentation . . . 842
12.316.2.1 ns . . . 842
12.316.2.2 nsmax . . . 843
12.316.2.3 nsta . . . 843
12.316.2.4 nt . . . 843
12.316.2.5 ntmax . . . 843
12.316.2.6 rr . . . 843
12.316.2.7 stas . . . 843
12.316.2.8 stec . . . 843
12.316.2.9 trop . . . 843
12.317 pps_tcp_rx Class Reference . . . 843
12.317.1 Detailed Description . . . 843
12.317.2 Member Data Documentation . . . 844
12.317.2.1 is_connected . . . 844
12.318 PpsSamplestamp Class Reference . . . 844
12.318.1 Detailed Description . . . 844
12.318.2 Member Data Documentation . . . 844
12.318.2.1 overflow_reg . . . 844
12.318.2.2 samplestamp . . . 844
12.319 prcopt_t Struct Reference . . . 844
12.319.1 Detailed Description . . . 845
12.319.2 Member Data Documentation . . . 845
12.319.2.1 antdel . . . 845
12.319.2.2 anttype . . . 846
12.319.2.3 arfilter . . . 846
12.319.2.4 armaxiter . . . 846
12.319.2.5 armaxposvar . . . 846
12.319.2.6 bancroft_init . . . 846
12.319.2.7 baseline . . . 846
12.319.2.8 bdsmodear . . . 846
12.319.2.9 codesmooth . . . 846
12.319.2.10 dynamics . . . 846

12.319.2.11 elmaskar . . . 846
12.319.2.12 elmaskhold . . . 846
12.319.2.13 elmin . . . 846
12.319.2.14 eratio . . . 847
12.319.2.15 err . . . 847
12.319.2.16 estqzsisb . . . 847
12.319.2.17 exsats . . . 847
12.319.2.18 exterr . . . 847
12.319.2.19 freqopt . . . 847
12.319.2.20 glomodear . . . 847
12.319.2.21 initrst . . . 847
12.319.2.22 intpref . . . 847
12.319.2.23 ionoopt . . . 847
12.319.2.24 maxaveep . . . 847
12.319.2.25 maxgdop . . . 847
12.319.2.26 maxinno . . . 848
12.319.2.27 maxout . . . 848
12.319.2.28 maxtdiff . . . 848
12.319.2.29 mindropsats . . . 848
12.319.2.30 minfix . . . 848
12.319.2.31 minfixsats . . . 848
12.319.2.32 minholdsats . . . 848
12.319.2.33 minlock . . . 848
12.319.2.34 mode . . . 848
12.319.2.35 modear . . . 848
12.319.2.36 navsys . . . 848
12.319.2.37 nf . . . 848
12.319.2.38 niter . . . 849
12.319.2.39 odisp . . . 849
12.319.2.40 outsingle . . . 849
12.319.2.41 pcvr . . . 849
12.319.2.42 posopt . . . 849
12.319.2.43 pppopt . . . 849
12.319.2.44 prn . . . 849
12.319.2.45 rb . . . 849
12.319.2.46 refpos . . . 849
12.319.2.47 rnxopt . . . 849
12.319.2.48 rovpos . . . 849
12.319.2.49 ru . . . 849
12.319.2.50 sateph . . . 850
12.319.2.51 sbascorr . . . 850
12.319.2.52 sbassatsel . . . 850

12.319.2.53 sclkstab . . . 850
12.319.2.54 snrmask . . . 850
12.319.2.55 soltype . . . 850
12.319.2.56 std . . . 850
12.319.2.57 syncsol . . . 850
12.319.2.58 thresar . . . 850
12.319.2.59 thresdop . . . 850
12.319.2.60 thresslip . . . 850
12.319.2.61 tidecorr . . . 850
12.319.2.62 tropopt . . . 851
12.319.2.63 varholdamb . . . 851
12.320 Protobuf_Cleanup_Manager Class Reference . . . 851
12.320.1 Detailed Description . . . 851
12.320.2 Member Function Documentation . . . 851
12.320.2.1 get() . . . 851
12.321 Osnma_Merkle_Tree_Material::PublicKeyEntry Struct Reference . . . 851
12.321.1 Detailed Description . . . 852
12.321.2 Member Data Documentation . . . 852
12.321.2.1 key_type . . . 852
12.321.2.2 leaf_index . . . 852
12.321.2.3 length_bits . . . 852
12.321.2.4 pkid . . . 852
12.321.2.5 pkid_valid . . . 852
12.321.2.6 point . . . 852
12.322 pulse_blanking_cc Class Reference . . . 852
12.322.1 Detailed Description . . . 853
12.323 PulseBlankingFilter Class Reference . . . 853
12.323.1 Detailed Description . . . 853
12.323.2 Member Function Documentation . . . 853
12.323.2.1 connect() . . . 853
12.323.2.2 disconnect() . . . 853
12.323.2.3 get_left_block() . . . 854
12.323.2.4 get_right_block() . . . 854
12.323.2.5 implementation() . . . 854
12.323.2.6 item_size() . . . 854
12.323.2.7 role() . . . 854
12.324 Pvt_Conf Class Reference . . . 854
12.324.1 Detailed Description . . . 856
12.324.2 Member Data Documentation . . . 856
12.324.2.1 an_dump_devname . . . 856
12.324.2.2 an_output_enabled . . . 856
12.324.2.3 an_rate_ms . . . 856

12.324.2.4 display_rate_ms . . . 856
12.324.2.5 dump . . . 856
12.324.2.6 dump_filename . . . 856
12.324.2.7 dump_mat . . . 856
12.324.2.8 enable_pvt_kf . . . 856
12.324.2.9 enable_rx_clock_correction . . . 856
12.324.2.10 flag_nmea_tty_port . . . 857
12.324.2.11 flag_rtcm_server . . . 857
12.324.2.12 flag_rtcm_tty_port . . . 857
12.324.2.13 geojson_output_enabled . . . 857
12.324.2.14 geojson_output_path . . . 857
12.324.2.15 geojson_rate_ms . . . 857
12.324.2.16 glonass_strict_health . . . 857
12.324.2.17 gpx_output_enabled . . . 857
12.324.2.18 gpx_output_path . . . 857
12.324.2.19 gpx_rate_ms . . . 857
12.324.2.20 has_output_file_path . . . 857
12.324.2.21 kf_use_imu_vel . . . 857
12.324.2.22 kml_output_enabled . . . 858
12.324.2.23 kml_output_path . . . 858
12.324.2.24 kml_rate_ms . . . 858
12.324.2.25 log_source_timetag . . . 858
12.324.2.26 log_source_timetag_file . . . 858
12.324.2.27 max_obs_block_rx_clock_offset_ms . . . 858
12.324.2.28 measures_ecef_pos_sd_m . . . 858
12.324.2.29 measures_ecef_vel_sd_ms . . . 858
12.324.2.30 monitor_enabled . . . 858
12.324.2.31 monitor_ephemeris_enabled . . . 858
12.324.2.32 nmea_dump_devname . . . 858
12.324.2.33 nmea_dump_filename . . . 858
12.324.2.34 nmea_output_file_enabled . . . 859
12.324.2.35 nmea_output_file_path . . . 859
12.324.2.36 nmea_rate_ms . . . 859
12.324.2.37 ntrip_caster_address . . . 859
12.324.2.38 ntrip_client_enabled . . . 859
12.324.2.39 ntrip_fallback_to_single . . . 859
12.324.2.40 ntrip_gga_period_ms . . . 859
12.324.2.41 ntrip_max_correction_age_s . . . 859
12.324.2.42 ntrip_mountpoint . . . 859
12.324.2.43 ntrip_password . . . 859
12.324.2.44 ntrip_password_env . . . 859
12.324.2.45 ntrip_port . . . 859

12.324.2.46 ntrip_reconnect_interval_ms . . . 860
12.324.2.47 ntrip_send_gga . . . 860
12.324.2.48 ntrip_station_id . . . 860
12.324.2.49 ntrip_timeout_ms . . . 860
12.324.2.50 ntrip_tls_enabled . . . 860
12.324.2.51 ntrip_username . . . 860
12.324.2.52 ntrip_version . . . 860
12.324.2.53 observable_interval_ms . . . 860
12.324.2.54 osnma_strict . . . 860
12.324.2.55 output_enabled . . . 860
12.324.2.56 output_path . . . 860
12.324.2.57 output_rate_ms . . . 860
12.324.2.58 protobuf_enabled . . . 861
12.324.2.59 ref_gps_week . . . 861
12.324.2.60 rinex_name . . . 861
12.324.2.61 rinex_output_enabled . . . 861
12.324.2.62 rinex_output_path . . . 861
12.324.2.63 rinex_version . . . 861
12.324.2.64 rinexobs_rate_ms . . . 861
12.324.2.65 rtcm_dump_devname . . . 861
12.324.2.66 rtcm_msg_rate_ms . . . 861
12.324.2.67 rtcm_output_file_enabled . . . 861
12.324.2.68 rtcm_output_file_path . . . 861
12.324.2.69 rtcm_station_id . . . 861
12.324.2.70 rtcm_tcp_port . . . 862
12.324.2.71 rtk_trace_level . . . 862
12.324.2.72 show_local_time_zone . . . 862
12.324.2.73 signal_enabled_flags . . . 862
12.324.2.74 system_ecef_pos_sd_m . . . 862
12.324.2.75 system_ecef_vel_sd_ms . . . 862
12.324.2.76 udp_addresses . . . 862
12.324.2.77 udp_eph_addresses . . . 862
12.324.2.78 udp_eph_port . . . 862
12.324.2.79 udp_ports . . . 862
12.324.2.80 use_e6_for_pvt . . . 862
12.324.2.81 use_has_corrections . . . 862
12.324.2.82 use_unhealthy_sats . . . 863
12.324.2.83 xml_output_enabled . . . 863
12.324.2.84 xml_output_path . . . 863
12.325 Pvt_Kf Class Reference . . . 863
12.325.1 Detailed Description . . . 863
12.326 Pvt_Solution Class Reference . . . 863

12.326.1 Detailed Description . . . 864
12.326.2 Member Function Documentation . . . 865
12.326.2.1 get_clock_drift_ppm() . . . 865
12.326.2.2 get_course_over_ground() . . . 865
12.326.2.3 get_height() . . . 865
12.326.2.4 get_latitude() . . . 865
12.326.2.5 get_longitude() . . . 865
12.326.2.6 get_num_valid_observations() . . . 865
12.326.2.7 get_ref_gps_week() . . . 865
12.326.2.8 get_speed_over_ground() . . . 865
12.326.2.9 get_time_offset_s() . . . 865
12.326.2.10 set_clock_drift_ppm() . . . 865
12.326.2.11 set_course_over_ground() . . . 865
12.326.2.12 set_num_valid_observations() . . . 866
12.326.2.13 set_ref_gps_week() . . . 866
12.326.2.14 set_rx_pos() . . . 866
12.326.2.15 set_rx_vel() . . . 866
12.326.2.16 set_speed_over_ground() . . . 866
12.326.2.17 set_time_offset_s() . . . 866
12.327 PvtInterface Class Reference . . . 866
12.327.1 Detailed Description . . . 867
12.327.2 Member Function Documentation . . . 867
12.327.2.1 get_beidou_cnav1_ephemeris() . . . 867
12.327.2.2 get_beidou_cnav2_ephemeris() . . . 867
12.328 Qzss_CNAV_Eop Class Reference . . . 868
12.328.1 Detailed Description . . . 868
12.328.2 Constructor & Destructor Documentation . . . 868
12.328.2.1 Qzss_CNAV_Eop() . . . 868
12.329 Qzss_CNAV_Iono Class Reference . . . 869
12.329.1 Detailed Description . . . 870
12.329.2 Constructor & Destructor Documentation . . . 870
12.329.2.1 Qzss_CNAV_Iono() [1/2] . . . 870
12.329.2.2 Qzss_CNAV_Iono() [2/2] . . . 870
12.330 Qzss_CNAV_Utc_Model Class Reference . . . 870
12.330.1 Detailed Description . . . 871
12.330.2 Constructor & Destructor Documentation . . . 871
12.330.2.1 Qzss_CNAV_Utc_Model() [1/2] . . . 871
12.330.2.2 Qzss_CNAV_Utc_Model() [2/2] . . . 871
12.331 Qzss_Iono Class Reference . . . 871
12.331.1 Detailed Description . . . 872
12.331.2 Constructor & Destructor Documentation . . . 873
12.331.2.1 Qzss_Iono() [1/2] . . . 873

12.331.2.2 Qzss_Iono() [2/2] . . . 873
12.332 Qzss_Utc_Model Class Reference . . . 873
12.332.1 Detailed Description . . . 874
12.332.2 Constructor & Destructor Documentation . . . 874
12.332.2.1 Qzss_Utc_Model() [1/2] . . . 874
12.332.2.2 Qzss_Utc_Model() [2/2] . . . 874
12.333 raw_t Struct Reference . . . 874
12.333.1 Detailed Description . . . 875
12.333.2 Member Data Documentation . . . 875
12.333.2.1 buff . . . 875
12.333.2.2 dpCA . . . 875
12.333.2.3 ephsat . . . 875
12.333.2.4 flag . . . 875
12.333.2.5 freqn . . . 875
12.333.2.6 halfc . . . 875
12.333.2.7 icpc . . . 875
12.333.2.8 icpp . . . 876
12.333.2.9 iod . . . 876
12.333.2.10 len . . . 876
12.333.2.11 lexmsg . . . 876
12.333.2.12 lockt . . . 876
12.333.2.13 msgtype . . . 876
12.333.2.14 nav . . . 876
12.333.2.15 nbyte . . . 876
12.333.2.16 obs . . . 876
12.333.2.17 obuf . . . 876
12.333.2.18 off . . . 876
12.333.2.19 opt . . . 876
12.333.2.20 outtype . . . 877
12.333.2.21 page . . . 877
12.333.2.22 pbuff . . . 877
12.333.2.23 pbyte . . . 877
12.333.2.24 plen . . . 877
12.333.2.25 prCA . . . 877
12.333.2.26 receive_time . . . 877
12.333.2.27 reply . . . 877
12.333.2.28 sbsmsg . . . 877
12.333.2.29 sta . . . 877
12.333.2.30 subfrm . . . 877
12.333.2.31 tbase . . . 877
12.333.2.32 time . . . 878
12.333.2.33 tobs . . . 878

12.333.2.34 tod . . . 878
12.333.2.35 week . . . 878
12.334 RawArraySignalSource Class Reference . . . 878
12.334.1 Detailed Description . . . 879
12.334.2 Member Function Documentation . . . 879
12.334.2.1 connect() . . . 879
12.334.2.2 disconnect() . . . 879
12.334.2.3 get_left_block() . . . 879
12.334.2.4 get_right_block() . . . 879
12.334.2.5 item_size() . . . 879
12.335 ReedSolomon Class Reference . . . 880
12.335.1 Detailed Description . . . 880
12.335.2 Constructor & Destructor Documentation . . . 880
12.335.2.1 ReedSolomon() [1/2] . . . 880
12.335.2.2 ReedSolomon() [2/2] . . . 880
12.335.3 Member Function Documentation . . . 881
12.335.3.1 decode() . . . 881
12.335.3.2 encode_with_generator_matrix() . . . 881
12.335.3.3 encode_with_generator_poly() . . . 881
12.336 Rinex_Printer Class Reference . . . 881
12.336.1 Detailed Description . . . 882
12.336.2 Constructor & Destructor Documentation . . . 882
12.336.2.1 Rinex_Printer() . . . 882
12.336.2.2 ~Rinex_Printer() . . . 882
12.336.3 Member Function Documentation . . . 882
12.336.3.1 get_navfilename() . . . 882
12.336.3.2 get_obsfilename() . . . 882
12.336.3.3 is_rinex_header_written() . . . 883
12.336.3.4 log_rinex_nav_bds_cnav1() . . . 883
12.336.3.5 log_rinex_nav_bds_cnav2() . . . 883
12.336.3.6 log_rinex_nav_bds_dnav() . . . 883
12.336.3.7 log_rinex_nav_gal_nav() . . . 883
12.336.3.8 log_rinex_nav_glo_gnav() . . . 883
12.336.3.9 log_rinex_nav_gps_cnav() . . . 883
12.336.3.10 log_rinex_nav_gps_nav() . . . 884
12.336.3.11 print_rinex_annotation() . . . 884
12.337 Rtcm Class Reference . . . 884
12.337.1 Detailed Description . . . 887
12.337.2 Constructor & Destructor Documentation . . . 887
12.337.2.1 Rtcm() . . . 887
12.337.3 Member Function Documentation . . . 888
12.337.3.1 bin_to_binary_data() . . . 888

12.337.3.2 bin_to_double() . . . 888
12.337.3.3 bin_to_hex() . . . 888
12.337.3.4 bin_to_uint() . . . 888
12.337.3.5 binary_data_to_bin() . . . 888
12.337.3.6 check_CRC() . . . 888
12.337.3.7 hex_to_bin() . . . 888
12.337.3.8 hex_to_int() . . . 889
12.337.3.9 hex_to_uint() . . . 889
12.337.3.10 is_server_running() . . . 889
12.337.3.11 lock_time() [1/4] . . . 889
12.337.3.12 lock_time() [2/4] . . . 889
12.337.3.13 lock_time() [3/4] . . . 890
12.337.3.14 lock_time() [4/4] . . . 890
12.337.3.15 print_IGM01() . . . 890
12.337.3.16 print_IGM02() . . . 890
12.337.3.17 print_IGM03() . . . 890
12.337.3.18 print_IGM05() . . . 890
12.337.3.19 print_MSM_1() . . . 890
12.337.3.20 print_MSM_2() . . . 891
12.337.3.21 print_MSM_3() . . . 891
12.337.3.22 print_MSM_4() . . . 891
12.337.3.23 print_MSM_5() . . . 892
12.337.3.24 print_MSM_6() . . . 892
12.337.3.25 print_MSM_7() . . . 892
12.337.3.26 print_MT1001() . . . 893
12.337.3.27 print_MT1002() . . . 893
12.337.3.28 print_MT1003() . . . 893
12.337.3.29 print_MT1004() . . . 893
12.337.3.30 print_MT1005() . . . 893
12.337.3.31 print_MT1005_test() . . . 894
12.337.3.32 print_MT1006() . . . 894
12.337.3.33 print_MT1008() . . . 894
12.337.3.34 print_MT1009() . . . 894
12.337.3.35 print_MT1010() . . . 895
12.337.3.36 print_MT1011() . . . 895
12.337.3.37 print_MT1012() . . . 896
12.337.3.38 print_MT1019() . . . 896
12.337.3.39 print_MT1020() . . . 897
12.337.3.40 print_MT1029() . . . 897
12.337.3.41 print_MT1045() . . . 897
12.337.3.42 print_MT1057() . . . 897
12.337.3.43 print_MT1058() . . . 897

12.337.3.44 print_MT1059() . . . 898
12.337.3.45 print_MT1060() . . . 898
12.337.3.46 read_MT1005() . . . 898
12.337.3.47 read_MT1019() . . . 898
12.337.3.48 read_MT1020() . . . 898
12.337.3.49 read_MT1045() . . . 899
12.337.3.50 run_server() . . . 899
12.337.3.51 send_message() . . . 899
12.337.3.52 stop_server() . . . 899
12.338 Rtcm_Printer Class Reference . . . 899
12.338.1 Detailed Description . . . 900
12.338.2 Constructor & Destructor Documentation . . . 900
12.338.2.1 Rtcm_Printer() . . . 900
12.338.2.2 ∼Rtcm_Printer() . . . 900
12.338.3 Member Function Documentation . . . 900
12.338.3.1 lock_time() . . . 900
12.338.3.2 print_MT1005_test() . . . 901
12.338.3.3 Print_Rtcm_Messages() . . . 901
12.339 rtcm_t Struct Reference . . . 901
12.339.1 Detailed Description . . . 902
12.339.2 Member Data Documentation . . . 902
12.339.2.1 buff . . . 902
12.339.2.2 cp . . . 902
12.339.2.3 dgps . . . 902
12.339.2.4 ephsat . . . 902
12.339.2.5 len . . . 902
12.339.2.6 lltime . . . 902
12.339.2.7 lock . . . 902
12.339.2.8 loss . . . 903
12.339.2.9 msg . . . 903
12.339.2.10 msgtype . . . 903
12.339.2.11 msmtype . . . 903
12.339.2.12 nav . . . 903
12.339.2.13 nbit . . . 903
12.339.2.14 nbyte . . . 903
12.339.2.15 nmsg2 . . . 903
12.339.2.16 nmsg3 . . . 903
12.339.2.17 obs . . . 903
12.339.2.18 obsflag . . . 903
12.339.2.19 opt . . . 903
12.339.2.20 outtype . . . 904
12.339.2.21 seqno . . . 904

12.339.2.22 ssr . . . 904
12.339.2.23 sta . . . 904
12.339.2.24 stah . . . 904
12.339.2.25 staid . . . 904
12.339.2.26 time . . . 904
12.339.2.27 time_s . . . 904
12.339.2.28 word . . . 904
12.340 rtk_t Struct Reference . . . 904
12.340.1 Detailed Description . . . 905
12.340.2 Member Data Documentation . . . 905
12.340.2.1 ambc . . . 905
12.340.2.2 errbuf . . . 905
12.340.2.3 excsat . . . 905
12.340.2.4 holdamb . . . 905
12.340.2.5 na . . . 905
12.340.2.6 nb_ar . . . 905
12.340.2.7 neb . . . 905
12.340.2.8 nfix . . . 905
12.340.2.9 nx . . . 906
12.340.2.10 opt . . . 906
12.340.2.11 P . . . 906
12.340.2.12 Pa . . . 906
12.340.2.13 rb . . . 906
12.340.2.14 sol . . . 906
12.340.2.15 ssat . . . 906
12.340.2.16 tt . . . 906
12.340.2.17 x . . . 906
12.340.2.18 xa . . . 906
12.341 Rtklib_Pvt Class Reference . . . 906
12.341.1 Detailed Description . . . 907
12.341.2 Member Function Documentation . . . 909
12.341.2.1 clear_ephemeris() . . . 909
12.341.2.2 clear_ephemeris_keep_almanac() . . . 909
12.341.2.3 connect() . . . 909
12.341.2.4 disconnect() . . . 909
12.341.2.5 get_beidou_cnav1_ephemeris() . . . 909
12.341.2.6 get_beidou_cnav2_ephemeris() . . . 909
12.341.2.7 get_beidou_dnav_almanac() . . . 909
12.341.2.8 get_beidou_dnav_ephemeris() . . . 910
12.341.2.9 get_galileo_almanac() . . . 910
12.341.2.10 get_galileo_ephemeris() . . . 910
12.341.2.11 get_glonass_almanac() . . . 910

12.341.2.12 get_glonass_ephemeris() . . . 910
12.341.2.13 get_glonass_utc_model() . . . 910
12.341.2.14 get_gps_almanac() . . . 910
12.341.2.15 get_gps_cnav_ephemeris() . . . 910
12.341.2.16 get_gps_ephemeris() . . . 910
12.341.2.17 get_latest_PVT() . . . 910
12.341.2.18 get_left_block() . . . 911
12.341.2.19 get_right_block() . . . 911
12.341.2.20 implementation() . . . 911
12.341.2.21 item_size() . . . 911
12.341.2.22 reset() . . . 911
12.341.2.23 role() . . . 911
12.342 rtklib_pvt_gs Class Reference . . . 911
12.342.1 Detailed Description . . . 912
12.342.2 Constructor & Destructor Documentation . . . 912
12.342.2.1 ~rtklib_pvt_gs() . . . 912
12.342.3 Member Function Documentation . . . 912
12.342.3.1 clear_ephemeris() . . . 912
12.342.3.2 clear_ephemeris_keep_almanac() . . . 913
12.342.3.3 get_beidou_dnav_almanac_map() . . . 913
12.342.3.4 get_beidou_dnav_ephemeris_map() . . . 913
12.342.3.5 get_galileo_almanac_map() . . . 913
12.342.3.6 get_galileo_ephemeris_map() . . . 913
12.342.3.7 get_gps_almanac_map() . . . 913
12.342.3.8 get_gps_cnav_ephemeris_map() . . . 913
12.342.3.9 get_latest_PVT() . . . 913
12.342.3.10 interpolate_observable() . . . 913
12.342.3.11 work() . . . 914
12.343 Rtklib_Solver Class Reference . . . 914
12.343.1 Detailed Description . . . 917
12.343.2 Member Function Documentation . . . 917
12.343.2.1 get_broadcast_signal_health() . . . 917
12.343.2.2 get_gdop() . . . 917
12.343.2.3 get_hdop() . . . 917
12.343.2.4 get_pdop() . . . 917
12.343.2.5 get_vdop() . . . 917
12.343.2.6 has_precise_ephemeris() . . . 918
12.343.2.7 solution_attempted() . . . 918
12.343.3 Member Data Documentation . . . 918
12.343.3.1 beidou_cnav1_ephemeris_map . . . 918
12.343.3.2 beidou_cnav1_iono . . . 918
12.343.3.3 beidou_cnav1_page_data_map . . . 918

12.343.3.4 beidou_cnav1_utc_model . . . 918
12.343.3.5 beidou_cnav2_ephemeris_map . . . 918
12.343.3.6 beidou_dnav_almanac_map . . . 918
12.343.3.7 beidou_dnav_ephemeris_map . . . 918
12.343.3.8 beidou_dnav_iono . . . 918
12.343.3.9 beidou_dnav_utc_model . . . 919
12.343.3.10 galileo_almanac_map . . . 919
12.343.3.11 galileo_ephemeris_map . . . 919
12.343.3.12 galileo_ephemeris_store . . . 919
12.343.3.13 galileo_iono . . . 919
12.343.3.14 galileo_reduced_ced_map . . . 919
12.343.3.15 galileo_utc_model . . . 919
12.343.3.16 glonass_gnav_almanac . . . 919
12.343.3.17 glonass_gnav_almanac_map . . . 919
12.343.3.18 glonass_gnav_ephemeris_map . . . 919
12.343.3.19 glonass_gnav_utc_model . . . 919
12.343.3.20 gps_almanac_map . . . 920
12.343.3.21 gps_cnav_eop . . . 920
12.343.3.22 gps_cnav_ephemeris_map . . . 920
12.343.3.23 gps_cnav_iono . . . 920
12.343.3.24 gps_cnav_utc_model . . . 920
12.343.3.25 gps_ephemeris_map . . . 920
12.343.3.26 gps_iono . . . 920
12.343.3.27 gps_previous_ephemeris_map . . . 920
12.343.3.28 gps_utc_model . . . 920
12.343.3.29 NTRIP_MIN_COMMON_SATELLITES . . . 920
12.343.3.30 pvt_sol . . . 920
12.343.3.31 pvt_ssat . . . 921
12.343.3.32 qzss_cnav_eop . . . 921
12.343.3.33 qzss_cnav_iono . . . 921
12.343.3.34 qzss_cnav_utc_model . . . 921
12.343.3.35 qzss_iono . . . 921
12.343.3.36 qzss_utc_model . . . 921
12.344 Rtklib_Tls_Client Class Reference . . . 921
12.344.1 Detailed Description . . . 921
12.344.2 Friends And Related Symbol Documentation . . . 922
12.344.2.1 Rtklib_Tls_Client_Test_Access . . . 922
12.345 rtksvr_t Struct Reference . . . 922
12.345.1 Detailed Description . . . 922
12.345.2 Member Data Documentation . . . 922
12.345.2.1 buff . . . 922
12.345.2.2 buffsize . . . 923

12.345.2.3 cputime . . . 923
12.345.2.4 cycle . . . 923
12.345.2.5 files . . . 923
12.345.2.6 format . . . 923
12.345.2.7 ftime . . . 923
12.345.2.8 lock . . . 923
12.345.2.9 moni . . . 923
12.345.2.10 nav . . . 923
12.345.2.11 navsel . . . 923
12.345.2.12 nb . . . 923
12.345.2.13 nmeacycle . . . 923
12.345.2.14 nmeapos . . . 924
12.345.2.15 nmeareq . . . 924
12.345.2.16 nmsg . . . 924
12.345.2.17 npb . . . 924
12.345.2.18 nsb . . . 924
12.345.2.19 nsbs . . . 924
12.345.2.20 nsol . . . 924
12.345.2.21 obs . . . 924
12.345.2.22 pbuf . . . 924
12.345.2.23 prcout . . . 924
12.345.2.24 raw . . . 924
12.345.2.25 rtcm . . . 924
12.345.2.26 rtk . . . 925
12.345.2.27 sbsmsg . . . 925
12.345.2.28 sbuf . . . 925
12.345.2.29 solbuf . . . 925
12.345.2.30 solopt . . . 925
12.345.2.31 state . . . 925
12.345.2.32 stream . . . 925
12.345.2.33 thread . . . 925
12.345.2.34 tick . . . 925
12.346 Rtl_Tcp_Dongle_Info Class Reference . . . 925
12.346.1 Detailed Description . . . 926
12.346.2 Member Enumeration Documentation . . . 926
12.346.2.1 anonymous enum . . . 926
12.346.3 Member Function Documentation . . . 926
12.346.3.1 get_tuner_gain_count() . . . 926
12.346.3.2 get_tuner_type() . . . 926
12.347 rtl_tcp_signal_source_c Class Reference . . . 926
12.347.1 Detailed Description . . . 927
12.348 RtlTcpSignalSource Class Reference . . . 927

12.348.1 Detailed Description . . . 928
12.348.2 Member Function Documentation . . . 928
12.348.2.1 connect() . . . 928
12.348.2.2 disconnect() . . . 928
12.348.2.3 get_left_block() . . . 928
12.348.2.4 get_right_block() . . . 928
12.348.2.5 item_size() . . . 928
12.349 SatelliteVisibility Class Reference . . . 928
12.349.1 Detailed Description . . . 929
12.349.2 Member Function Documentation . . . 929
12.349.2.1 enabled() . . . 929
12.349.2.2 search_ratio() . . . 929
12.349.2.3 SetCommandReference() . . . 929
12.349.2.4 Tick() . . . 929
12.350 Sbas_Ephemeris Class Reference . . . 930
12.350.1 Detailed Description . . . 930
12.350.2 Member Data Documentation . . . 930
12.350.2.1 b_sv_do_not_use . . . 930
12.350.2.2 d_acc . . . 931
12.350.2.3 d_af0 . . . 931
12.350.2.4 d_af1 . . . 931
12.350.2.5 d_pos . . . 931
12.350.2.6 d_tof . . . 931
12.350.2.7 d_vel . . . 931
12.350.2.8 i_prn . . . 931
12.350.2.9 i_sv_ura . . . 931
12.350.2.10 i_t0 . . . 931
12.351 sbas_l1_telemetry_decoder_gs Class Reference . . . 932
12.351.1 Detailed Description . . . 933
12.351.2 Member Function Documentation . . . 933
12.351.2.1 consume_decoded_timestamps() . . . 933
12.351.2.2 general_work() . . . 933
12.351.2.3 reset() . . . 933
12.351.2.4 set_channel() . . . 933
12.351.2.5 set_satellite() . . . 933
12.352 Sbas_Raw_Message Class Reference . . . 933
12.352.1 Detailed Description . . . 934
12.352.2 Constructor & Destructor Documentation . . . 934
12.352.2.1 Sbas_Raw_Message() . . . 934
12.352.3 Member Function Documentation . . . 934
12.352.3.1 frame() . . . 934
12.352.3.2 message_type() . . . 934

12.352.3.3 prn() . . . 934
12.352.3.4 sample_stamp_s() . . . 934
12.353 Sbas_Rtklib_Corrections Class Reference . . . 934
12.353.1 Detailed Description . . . 935
12.353.2 Member Function Documentation . . . 935
12.353.2.1 active_prn() . . . 935
12.353.2.2 ionosphere_corrections() . . . 935
12.353.2.3 pending_messages() . . . 935
12.353.2.4 satellite_corrections() . . . 935
12.354 SbasL1DumpSession Class Reference . . . 935
12.354.1 Detailed Description . . . 935
12.354.2 Member Function Documentation . . . 935
12.354.2.1 open() . . . 935
12.355 sbs_t Struct Reference . . . 936
12.355.1 Detailed Description . . . 936
12.355.2 Member Data Documentation . . . 936
12.355.2.1 msgs . . . 936
12.355.2.2 n . . . 936
12.355.2.3 nmax . . . 936
12.356 sbsfcorr_t Struct Reference . . . 936
12.356.1 Detailed Description . . . 936
12.356.2 Member Data Documentation . . . 936
12.356.2.1 ai . . . 936
12.356.2.2 dt . . . 936
12.356.2.3 iodf . . . 937
12.356.2.4 prc . . . 937
12.356.2.5 rrc . . . 937
12.356.2.6 t0 . . . 937
12.356.2.7 udre . . . 937
12.357 sbsigp_t Struct Reference . . . 937
12.357.1 Detailed Description . . . 937
12.357.2 Member Data Documentation . . . 937
12.357.2.1 delay . . . 937
12.357.2.2 give . . . 937
12.357.2.3 lat . . . 937
12.357.2.4 lon . . . 938
12.357.2.5 t0 . . . 938
12.358 sbsigpband_t Struct Reference . . . 938
12.358.1 Detailed Description . . . 938
12.358.2 Member Data Documentation . . . 938
12.358.2.1 bite . . . 938
12.358.2.2 bits . . . 938

12.358.2.3 x . . . 938
12.358.2.4 y . . . 938
12.359 sbsion_t Struct Reference . . . 938
12.359.1 Detailed Description . . . 939
12.359.2 Member Data Documentation . . . 939
12.359.2.1 igp . . . 939
12.359.2.2 iodi . . . 939
12.359.2.3 nigp . . . 939
12.360 sbslcorr_t Struct Reference . . . 939
12.360.1 Detailed Description . . . 939
12.360.2 Member Data Documentation . . . 939
12.360.2.1 daf0 . . . 939
12.360.2.2 daf1 . . . 939
12.360.2.3 dpos . . . 939
12.360.2.4 dvel . . . 939
12.360.2.5 iode . . . 940
12.360.2.6 t0 . . . 940
12.361 sbsmsg_t Struct Reference . . . 940
12.361.1 Detailed Description . . . 940
12.361.2 Member Data Documentation . . . 940
12.361.2.1 msg . . . 940
12.361.2.2 prn . . . 940
12.361.2.3 tow . . . 940
12.361.2.4 week . . . 940
12.362 sbssat_t Struct Reference . . . 940
12.362.1 Detailed Description . . . 941
12.362.2 Member Data Documentation . . . 941
12.362.2.1 iodp . . . 941
12.362.2.2 nsat . . . 941
12.362.2.3 sat . . . 941
12.362.2.4 tlat . . . 941
12.363 sbssatp_t Struct Reference . . . 941
12.363.1 Detailed Description . . . 941
12.363.2 Member Data Documentation . . . 941
12.363.2.1 fcorr . . . 941
12.363.2.2 lcorr . . . 941
12.363.2.3 sat . . . 941
12.364 sdr_gnss_packet_t Struct Reference . . . 942
12.364.1 Detailed Description . . . 942
12.364.2 Member Data Documentation . . . 942
12.364.2.1 galileo_satellites . . . 942
12.364.2.2 gps_satellites . . . 942

12.364.2.3 height . . . 942
12.364.2.4 latitude . . . 942
12.364.2.5 longitude . . . 942
12.364.2.6 microseconds . . . 942
12.364.2.7 nsvfix . . . 942
12.364.2.8 reserved . . . 943
12.364.2.9 status . . . 943
12.364.2.10 velocity . . . 943
12.365 Secure_String Class Reference . . . 943
12.365.1 Detailed Description . . . 943
12.365.2 Constructor & Destructor Documentation . . . 943
12.365.2.1 Secure_String() [1/3] . . . 943
12.365.2.2 Secure_String() [2/3] . . . 943
12.365.2.3 Secure_String() [3/3] . . . 944
12.365.2.4 ~Secure_String() . . . 944
12.365.3 Member Function Documentation . . . 944
12.365.3.1 clear() . . . 944
12.365.3.2 empty() . . . 944
12.365.3.3 operator=() [1/3] . . . 944
12.365.3.4 operator=() [2/3] . . . 944
12.365.3.5 operator=() [3/3] . . . 944
12.365.3.6 value() . . . 944
12.366 IONGSMSFileSource::SegmentDescriptor Struct Reference . . . 944
12.366.1 Detailed Description . . . 945
12.366.2 Member Data Documentation . . . 945
12.366.2.1 block . . . 945
12.366.2.2 block_extends_to_eof . . . 945
12.366.2.3 block_start_offset . . . 945
12.366.2.4 data_filepath . . . 945
12.367 SensorDataAggregator Class Reference . . . 945
12.367.1 Detailed Description . . . 945
12.367.2 Member Data Documentation . . . 945
12.367.2.1 SENSOR_DATA_TAG . . . 945
12.368 SensorDataConfiguration Struct Reference . . . 946
12.368.1 Detailed Description . . . 946
12.368.2 Member Data Documentation . . . 946
12.368.2.1 file_id . . . 946
12.368.2.2 id . . . 946
12.368.2.3 identifier . . . 946
12.368.2.4 offset . . . 946
12.368.2.5 tag_key . . . 946
12.368.2.6 type . . . 946

12.369 SensorDataFile Class Reference . . . 946
12.369.1 Detailed Description . . . 947
12.369.2 Member Typedef Documentation . . . 947
12.369.2.1 id_type . . . 947
12.369.2.2 sptr . . . 947
12.370 SensorDataFileConfiguration Struct Reference . . . 947
12.370.1 Detailed Description . . . 947
12.370.2 Member Data Documentation . . . 947
12.370.2.1 chunk_size . . . 947
12.370.2.2 file_offset . . . 947
12.370.2.3 filename . . . 947
12.370.2.4 id . . . 948
12.370.2.5 repeat . . . 948
12.370.2.6 sample_offset . . . 948
12.370.2.7 sample_period . . . 948
12.371 SensorDataSample< DataType > Struct Template Reference . . . 948
12.371.1 Detailed Description . . . 948
12.371.2 Constructor & Destructor Documentation . . . 948
12.371.2.1 SensorDataSample() . . . 948
12.371.3 Member Data Documentation . . . 948
12.371.3.1 timestamp . . . 948
12.371.3.2 value . . . 949
12.372 SensorDataSource Class Reference . . . 949
12.372.1 Detailed Description . . . 949
12.372.2 Member Typedef Documentation . . . 949
12.372.2.1 sptr . . . 949
12.373 SensorDataSourceConfiguration Class Reference . . . 949
12.373.1 Detailed Description . . . 950
12.374 SensorDataType Struct Reference . . . 950
12.374.1 Detailed Description . . . 950
12.374.2 Member Enumeration Documentation . . . 950
12.374.2.1 value_type . . . 950
12.375 SensorIdentifier Struct Reference . . . 950
12.375.1 Detailed Description . . . 950
12.375.2 Member Enumeration Documentation . . . 951
12.375.2.1 value_type . . . 951
12.376 seph_t Struct Reference . . . 951
12.376.1 Detailed Description . . . 951
12.376.2 Member Data Documentation . . . 951
12.376.2.1 acc . . . 951
12.376.2.2 af0 . . . 951
12.376.2.3 af1 . . . 951

12.376.2.4 pos . . . 951
12.376.2.5 sat . . . 951
12.376.2.6 sva . . . 951
12.376.2.7 svh . . . 952
12.376.2.8 t0 . . . 952
12.376.2.9 tof . . . 952
12.376.2.10 vel . . . 952
12.377 Serdes_Galileo_Eph Class Reference . . . 952
12.377.1 Detailed Description . . . 952
12.377.2 Constructor & Destructor Documentation . . . 952
12.377.2.1 Serdes_Galileo_Eph() [1/3] . . . 952
12.377.2.2 Serdes_Galileo_Eph() [2/3] . . . 952
12.377.2.3 Serdes_Galileo_Eph() [3/3] . . . 953
12.377.3 Member Function Documentation . . . 953
12.377.3.1 createProtobuffer() . . . 953
12.377.3.2 operator=() [1/2] . . . 953
12.377.3.3 operator=() [2/2] . . . 953
12.377.3.4 readProtobuffer() . . . 953
12.378 Serdes_Gnss_Synchro Class Reference . . . 953
12.378.1 Detailed Description . . . 954
12.378.2 Constructor & Destructor Documentation . . . 954
12.378.2.1 Serdes_Gnss_Synchro() [1/3] . . . 954
12.378.2.2 Serdes_Gnss_Synchro() [2/3] . . . 954
12.378.2.3 Serdes_Gnss_Synchro() [3/3] . . . 954
12.378.3 Member Function Documentation . . . 954
12.378.3.1 createProtobuffer() . . . 954
12.378.3.2 operator=() [1/2] . . . 954
12.378.3.3 operator=() [2/2] . . . 955
12.378.3.4 readProtobuffer() . . . 955
12.379 Serdes_Gps_Eph Class Reference . . . 955
12.379.1 Detailed Description . . . 955
12.379.2 Constructor & Destructor Documentation . . . 955
12.379.2.1 Serdes_Gps_Eph() [1/3] . . . 955
12.379.2.2 Serdes_Gps_Eph() [2/3] . . . 956
12.379.2.3 Serdes_Gps_Eph() [3/3] . . . 956
12.379.3 Member Function Documentation . . . 956
12.379.3.1 createProtobuffer() . . . 956
12.379.3.2 operator=() [1/2] . . . 956
12.379.3.3 operator=() [2/2] . . . 956
12.379.3.4 readProtobuffer() . . . 956
12.380 Serdes_Monitor_Pvt Class Reference . . . 957
12.380.1 Detailed Description . . . 957

12.380.2 Constructor & Destructor Documentation . . . 957
12.380.2.1 Serdes_Monitor_Pvt() [1/3] . . . 957
12.380.2.2 Serdes_Monitor_Pvt() [2/3] . . . 957
12.380.2.3 Serdes_Monitor_Pvt() [3/3] . . . 957
12.380.3 Member Function Documentation . . . 957
12.380.3.1 createProtobuffer() . . . 957
12.380.3.2 operator=() [1/2] . . . 958
12.380.3.3 operator=() [2/2] . . . 958
12.380.3.4 readProtobuffer() . . . 958
12.381 Serdes_Nav_Message Class Reference . . . 958
12.381.1 Detailed Description . . . 958
12.381.2 Constructor & Destructor Documentation . . . 958
12.381.2.1 Serdes_Nav_Message() [1/3] . . . 958
12.381.2.2 Serdes_Nav_Message() [2/3] . . . 958
12.381.2.3 Serdes_Nav_Message() [3/3] . . . 959
12.381.3 Member Function Documentation . . . 959
12.381.3.1 createProtobuffer() . . . 959
12.381.3.2 operator=() [1/2] . . . 959
12.381.3.3 operator=() [2/2] . . . 959
12.381.3.4 readProtobuffer() . . . 959
12.382 serial_t Struct Reference . . . 959
12.382.1 Detailed Description . . . 959
12.382.2 Member Data Documentation . . . 960
12.382.2.1 dev . . . 960
12.382.2.2 error . . . 960
12.383 short_x2_to_cshort Class Reference . . . 960
12.383.1 Detailed Description . . . 960
12.384 Signal_Enabled_Flags Class Reference . . . 960
12.384.1 Detailed Description . . . 961
12.384.2 Member Function Documentation . . . 961
12.384.2.1 check_any_enabled() . . . 961
12.384.2.2 check_only_enabled() . . . 961
12.384.2.3 or_all() [1/2] . . . 961
12.384.2.4 or_all() [2/2] . . . 961
12.384.3 Member Data Documentation . . . 961
12.384.3.1 flags . . . 961
12.384.3.2 has_beidou . . . 961
12.384.3.3 has_galileo . . . 962
12.384.3.4 has_glonass . . . 962
12.384.3.5 has_gps . . . 962
12.384.3.6 has_qzss . . . 962
12.384.3.7 only_beidou . . . 962

12.384.3.8 only_galileo . . . 962
12.384.3.9 only_glonass . . . 962
12.384.3.10 only_gps . . . 962
12.384.3.11 only_qzss . . . 962
12.385 signal_generator_c Class Reference . . . 962
12.385.1 Detailed Description . . . 963
12.385.2 Friends And Related Symbol Documentation . . . 963
12.385.2.1 signal_make_generator_c . . . 963
12.386 SignalConditioner Class Reference . . . 963
12.386.1 Detailed Description . . . 964
12.386.2 Constructor & Destructor Documentation . . . 964
12.386.2.1 SignalConditioner() . . . 964
12.386.2.2 ~SignalConditioner() . . . 964
12.386.3 Member Function Documentation . . . 964
12.386.3.1 connect() . . . 964
12.386.3.2 data_type_adapter() . . . 965
12.386.3.3 disconnect() . . . 965
12.386.3.4 get_left_block() . . . 965
12.386.3.5 get_right_block() . . . 965
12.386.3.6 implementation() . . . 965
12.386.3.7 input_filter() . . . 965
12.386.3.8 item_size() . . . 965
12.386.3.9 resampler() . . . 965
12.386.3.10 role() . . . 965
12.387 SignalGenerator Class Reference . . . 965
12.387.1 Detailed Description . . . 966
12.387.2 Member Function Documentation . . . 966
12.387.2.1 connect() . . . 966
12.387.2.2 disconnect() . . . 966
12.387.2.3 get_left_block() . . . 966
12.387.2.4 get_right_block() . . . 966
12.387.2.5 implementation() . . . 967
12.387.2.6 item_size() . . . 967
12.387.2.7 role() . . . 967
12.388 SignalSourceBase Class Reference . . . 967
12.388.1 Detailed Description . . . 968
12.388.2 Constructor & Destructor Documentation . . . 968
12.388.2.1 SignalSourceBase() . . . 968
12.388.3 Member Function Documentation . . . 968
12.388.3.1 decode_item_type() . . . 968
12.388.3.2 get_left_block() . . . 969
12.388.3.3 getRfChannels() . . . 969

12.388.3.4 implementation() . . . 969
12.388.3.5 role() . . . 969
12.389 SignalSourceInterface Class Reference . . . 969
12.389.1 Detailed Description . . . 970
12.390 snrmask_t Struct Reference . . . 971
12.390.1 Detailed Description . . . 971
12.390.2 Member Data Documentation . . . 971
12.390.2.1 ena . . . 971
12.390.2.2 mask . . . 971
12.391 sol_t Struct Reference . . . 971
12.391.1 Detailed Description . . . 971
12.391.2 Member Data Documentation . . . 971
12.391.2.1 age . . . 971
12.391.2.2 dtr . . . 971
12.391.2.3 ns . . . 972
12.391.2.4 prev_ratio1 . . . 972
12.391.2.5 prev_ratio2 . . . 972
12.391.2.6 qr . . . 972
12.391.2.7 ratio . . . 972
12.391.2.8 rr . . . 972
12.391.2.9 stat . . . 972
12.391.2.10 thres . . . 972
12.391.2.11 time . . . 972
12.391.2.12 type . . . 972
12.392 solbuf_t Struct Reference . . . 972
12.392.1 Detailed Description . . . 973
12.392.2 Member Data Documentation . . . 973
12.392.2.1 buff . . . 973
12.392.2.2 cyclic . . . 973
12.392.2.3 data . . . 973
12.392.2.4 end . . . 973
12.392.2.5 n . . . 973
12.392.2.6 nb . . . 973
12.392.2.7 nmax . . . 973
12.392.2.8 rb . . . 973
12.392.2.9 start . . . 973
12.392.2.10 time . . . 974
12.393 solopt_t Struct Reference . . . 974
12.393.1 Detailed Description . . . 974
12.393.2 Member Data Documentation . . . 974
12.393.2.1 datum . . . 974
12.393.2.2 degf . . . 974

12.393.2.3 geoid . . . 974
12.393.2.4 height . . . 974
12.393.2.5 maxsolstd . . . 974
12.393.2.6 nmeaintv . . . 975
12.393.2.7 outhead . . . 975
12.393.2.8 outopt . . . 975
12.393.2.9 posf . . . 975
12.393.2.10 prog . . . 975
12.393.2.11 sep . . . 975
12.393.2.12 solstatic . . . 975
12.393.2.13 sstat . . . 975
12.393.2.14 timef . . . 975
12.393.2.15 times . . . 975
12.393.2.16 timeu . . . 975
12.393.2.17 trace . . . 975
12.394 solstat_t Struct Reference . . . 976
12.394.1 Detailed Description . . . 976
12.394.2 Member Data Documentation . . . 976
12.394.2.1 az . . . 976
12.394.2.2 el . . . 976
12.394.2.3 flag . . . 976
12.394.2.4 frq . . . 976
12.394.2.5 lock . . . 976
12.394.2.6 outc . . . 976
12.394.2.7 rejc . . . 976
12.394.2.8 resc . . . 977
12.394.2.9 resp . . . 977
12.394.2.10 sat . . . 977
12.394.2.11 slipc . . . 977
12.394.2.12 snr . . . 977
12.394.2.13 time . . . 977
12.395 solstatbuf_t Struct Reference . . . 977
12.395.1 Detailed Description . . . 977
12.395.2 Member Data Documentation . . . 977
12.395.2.1 data . . . 977
12.395.2.2 n . . . 977
12.395.2.3 nmax . . . 978
12.396 SpirFileSignalSource Class Reference . . . 978
12.396.1 Detailed Description . . . 980
12.396.2 Member Function Documentation . . . 980
12.396.2.1 create_file_source_hook() . . . 980
12.396.2.2 itemTypeToSize() . . . 980

12.396.2.3 post_disconnect_hook() . . . 980
12.396.2.4 pre_connect_hook() . . . 980
12.396.2.5 source() . . . 980
12.397 ssat_t Struct Reference . . . 980
12.397.1 Detailed Description . . . 981
12.397.2 Member Data Documentation . . . 981
12.397.2.1 azel . . . 981
12.397.2.2 code . . . 981
12.397.2.3 fix . . . 981
12.397.2.4 gf . . . 981
12.397.2.5 gf2 . . . 981
12.397.2.6 half . . . 981
12.397.2.7 lock . . . 981
12.397.2.8 mw . . . 982
12.397.2.9 outc . . . 982
12.397.2.10 ph . . . 982
12.397.2.11 phw . . . 982
12.397.2.12 pt . . . 982
12.397.2.13 rejc . . . 982
12.397.2.14 resc . . . 982
12.397.2.15 resp . . . 982
12.397.2.16 slip . . . 982
12.397.2.17 slipc . . . 982
12.397.2.18 snr . . . 982
12.397.2.19 snr_base . . . 982
12.397.2.20 snr_rover . . . 983
12.397.2.21 sys . . . 983
12.397.2.22 vs . . . 983
12.397.2.23 vsat . . . 983
12.398 ssr_t Struct Reference . . . 983
12.398.1 Detailed Description . . . 983
12.398.2 Member Data Documentation . . . 983
12.398.2.1 cbias . . . 983
12.398.2.2 dclk . . . 983
12.398.2.3 ddeph . . . 984
12.398.2.4 deph . . . 984
12.398.2.5 hrclk . . . 984
12.398.2.6 iod . . . 984
12.398.2.7 iodcrc . . . 984
12.398.2.8 iode . . . 984
12.398.2.9 pbias . . . 984
12.398.2.10 refd . . . 984

12.398.2.11 stdpb . . . 984
12.398.2.12 t0 . . . 984
12.398.2.13 udi . . . 984
12.398.2.14 update . . . 984
12.398.2.15 ura . . . 985
12.398.2.16 yaw_ang . . . 985
12.398.2.17 yaw_rate . . . 985
12.399 sta_t Struct Reference . . . 985
12.399.1 Detailed Description . . . 985
12.399.2 Member Data Documentation . . . 985
12.399.2.1 antdes . . . 985
12.399.2.2 antsetup . . . 985
12.399.2.3 antsno . . . 985
12.399.2.4 del . . . 986
12.399.2.5 deltype . . . 986
12.399.2.6 hgt . . . 986
12.399.2.7 itrf . . . 986
12.399.2.8 marker . . . 986
12.399.2.9 name . . . 986
12.399.2.10 pos . . . 986
12.399.2.11 recsno . . . 986
12.399.2.12 rectype . . . 986
12.399.2.13 recver . . . 986
12.400 stec_t Struct Reference . . . 986
12.400.1 Detailed Description . . . 987
12.400.2 Member Data Documentation . . . 987
12.400.2.1 azel . . . 987
12.400.2.2 flag . . . 987
12.400.2.3 ion . . . 987
12.400.2.4 sat . . . 987
12.400.2.5 std . . . 987
12.400.2.6 time . . . 987
12.401 stream_cfg Struct Reference . . . 987
12.401.1 Detailed Description . . . 987
12.401.2 Member Data Documentation . . . 988
12.401.2.1 bw_hz . . . 988
12.401.2.2 fs_hz . . . 988
12.401.2.3 lo_hz . . . 988
12.401.2.4 rfport . . . 988
12.402 stream_t Struct Reference . . . 988
12.402.1 Detailed Description . . . 988
12.402.2 Member Data Documentation . . . 988

12.402.2.1 inb . . . 988
12.402.2.2 inbt . . . 988
12.402.2.3 inr . . . 989
12.402.2.4 lock . . . 989
12.402.2.5 mode . . . 989
12.402.2.6 msg . . . 989
12.402.2.7 outb . . . 989
12.402.2.8 outbt . . . 989
12.402.2.9 outr . . . 989
12.402.2.10 path . . . 989
12.402.2.11 port . . . 989
12.402.2.12 state . . . 989
12.402.2.13 tact . . . 989
12.402.2.14 tick . . . 989
12.402.2.15 type . . . 990
12.403 StringConverter Class Reference . . . 990
12.403.1 Detailed Description . . . 990
12.404 Tag Class Reference . . . 990
12.404.1 Detailed Description . . . 991
12.404.2 Member Enumeration Documentation . . . 991
12.404.2.1 e_verification_status . . . 991
12.404.3 Constructor & Destructor Documentation . . . 991
12.404.3.1 Tag() [1/2] . . . 991
12.404.3.2 Tag() [2/2] . . . 991
12.404.4 Member Data Documentation . . . 991
12.404.4.1 ADKD . . . 991
12.404.4.2 allowed_adkds . . . 991
12.404.4.3 computed_tag . . . 991
12.404.4.4 cop . . . 992
12.404.4.5 CTR . . . 992
12.404.4.6 id_counter . . . 992
12.404.4.7 nav_data . . . 992
12.404.4.8 nmas . . . 992
12.404.4.9 PRN_d . . . 992
12.404.4.10 PRNa . . . 992
12.404.4.11 received_tag . . . 992
12.404.4.12 skipped . . . 992
12.404.4.13 status . . . 992
12.404.4.14 tag_id . . . 992
12.404.4.15 TOW . . . 992
12.404.4.16 WN . . . 993
12.405 Tcp_Communication Class Reference . . . 993

12.405.1 Detailed Description . . . 993
12.406 Tcp_Packet_Data Class Reference . . . 993
12.406.1 Detailed Description . . . 993
12.406.2 Member Data Documentation . . . 993
12.406.2.1 proc_pack_carr_error . . . 993
12.406.2.2 proc_pack_carrier_doppler_hz . . . 993
12.406.2.3 proc_pack_code_error . . . 994
12.407 tcp_t Struct Reference . . . 994
12.407.1 Detailed Description . . . 994
12.407.2 Member Data Documentation . . . 994
12.407.2.1 addr . . . 994
12.407.2.2 port . . . 994
12.407.2.3 saddr . . . 994
12.407.2.4 sock . . . 994
12.407.2.5 state . . . 994
12.407.2.6 tact . . . 994
12.407.2.7 tcon . . . 995
12.407.2.8 tdis . . . 995
12.408 tcpcli_t Struct Reference . . . 995
12.408.1 Detailed Description . . . 995
12.408.2 Member Data Documentation . . . 995
12.408.2.1 svr . . . 995
12.408.2.2 tirecon . . . 995
12.408.2.3 toinact . . . 995
12.409 TcpCmdInterface Class Reference . . . 995
12.409.1 Detailed Description . . . 996
12.409.2 Member Function Documentation . . . 996
12.409.2.1 get_LLH() . . . 996
12.409.2.2 get_utc_time() . . . 996
12.410 tcpsvr_t Struct Reference . . . 996
12.410.1 Detailed Description . . . 996
12.410.2 Member Data Documentation . . . 996
12.410.2.1 cli . . . 996
12.410.2.2 svr . . . 996
12.411 tec_t Struct Reference . . . 996
12.411.1 Detailed Description . . . 997
12.411.2 Member Data Documentation . . . 997
12.411.2.1 data . . . 997
12.411.2.2 hgts . . . 997
12.411.2.3 lats . . . 997
12.411.2.4 lons . . . 997
12.411.2.5 ndata . . . 997

12.411.2.6 rb . . . 997
12.411.2.7 rms . . . 997
12.411.2.8 time . . . 997
12.412 telemetry_impl_interface Class Reference . . . 997
12.412.1 Detailed Description . . . 998
12.412.2 Constructor & Destructor Documentation . . . 998
12.412.2.1 telemetry_impl_interface() . . . 998
12.412.3 Member Function Documentation . . . 999
12.412.3.1 reset() . . . 999
12.412.3.2 set_channel() . . . 999
12.412.3.3 set_satellite() . . . 999
12.413 TelemetryDecoderAdapter Class Reference . . . 999
12.413.1 Detailed Description . . . 1000
12.413.2 Member Function Documentation . . . 1000
12.413.2.1 connect() . . . 1000
12.413.2.2 disconnect() . . . 1000
12.413.2.3 get_left_block() . . . 1000
12.413.2.4 get_right_block() . . . 1000
12.413.2.5 implementation() . . . 1000
12.413.2.6 item_size() . . . 1000
12.413.2.7 reset() . . . 1000
12.413.2.8 role() . . . 1000
12.413.2.9 set_channel() . . . 1000
12.413.2.10 set_satellite() . . . 1000
12.414 TelemetryDecoderInterface Class Reference . . . 1001
12.414.1 Detailed Description . . . 1001
12.415 tle_t Struct Reference . . . 1001
12.415.1 Detailed Description . . . 1002
12.415.2 Member Data Documentation . . . 1002
12.415.2.1 data . . . 1002
12.415.2.2 n . . . 1002
12.415.2.3 nmax . . . 1002
12.416 tled_t Struct Reference . . . 1002
12.416.1 Detailed Description . . . 1002
12.416.2 Member Data Documentation . . . 1002
12.416.2.1 alias . . . 1002
12.416.2.2 bstar . . . 1003
12.416.2.3 desig . . . 1003
12.416.2.4 ecc . . . 1003
12.416.2.5 eleno . . . 1003
12.416.2.6 epoch . . . 1003
12.416.2.7 etype . . . 1003

12.416.2.8 inc . . . 1003
12.416.2.9 M . . . 1003
12.416.2.10 n . . . 1003
12.416.2.11 name . . . 1003
12.416.2.12 nddot . . . 1003
12.416.2.13 ndot . . . 1003
12.416.2.14 OMG . . . 1004
12.416.2.15 omg . . . 1004
12.416.2.16 rev . . . 1004
12.416.2.17 satclass . . . 1004
12.416.2.18 satno . . . 1004
12.417 Tlm_Conf Class Reference . . . 1004
12.417.1 Detailed Description . . . 1004
12.417.2 Member Data Documentation . . . 1004
12.417.2.1 dump . . . 1004
12.417.2.2 dump_crc_stats . . . 1005
12.417.2.3 dump_crc_stats_filename . . . 1005
12.417.2.4 dump_filename . . . 1005
12.417.2.5 dump_mat . . . 1005
12.417.2.6 enable_navdata_monitor . . . 1005
12.417.2.7 enable_reed_solomon . . . 1005
12.417.2.8 remove_dat . . . 1005
12.417.2.9 there_are_e1_channels . . . 1005
12.417.2.10 there_are_e6_channels . . . 1005
12.417.2.11 tow_to_trk . . . 1005
12.417.2.12 use_ced . . . 1005
12.418 Tlm_CRC_Stats Class Reference . . . 1006
12.418.1 Detailed Description . . . 1006
12.418.2 Member Function Documentation . . . 1006
12.418.2.1 initialize() . . . 1006
12.418.2.2 set_channel() . . . 1006
12.418.2.3 update_CRC_stats() . . . 1006
12.419 TOW_to_trk Class Reference . . . 1006
12.419.1 Detailed Description . . . 1007
12.419.2 Constructor & Destructor Documentation . . . 1007
12.419.2.1 TOW_to_trk() . . . 1007
12.419.3 Member Data Documentation . . . 1007
12.419.3.1 channel . . . 1007
12.419.3.2 prn . . . 1007
12.419.3.3 sample_stamp . . . 1007
12.419.3.4 signal . . . 1007
12.419.3.5 tow . . . 1007

12.419.3.6 wn . . . 1007
12.420 Monitor_Pvt::TrackedSatelliteInfo Class Reference . . . 1007
12.420.1 Detailed Description . . . 1008
12.420.2 Member Function Documentation . . . 1008
12.420.2.1 serialize() . . . 1008
12.420.3 Member Data Documentation . . . 1008
12.420.3.1 azimuth_deg . . . 1008
12.420.3.2 combined . . . 1008
12.420.3.3 elevation_deg . . . 1008
12.420.3.4 healthy . . . 1008
12.420.3.5 prn . . . 1008
12.420.3.6 signal . . . 1009
12.420.3.7 system . . . 1009
12.420.3.8 used . . . 1009
12.421 Tracking_2nd_DLL_filter Class Reference . . . 1009
12.421.1 Detailed Description . . . 1009
12.421.2 Member Function Documentation . . . 1009
12.421.2.1 get_code_nco() . . . 1009
12.421.2.2 initialize() . . . 1009
12.421.2.3 set_DLL_BW() . . . 1010
12.421.2.4 set_pdi() . . . 1010
12.422 Tracking_2nd_PLL_filter Class Reference . . . 1010
12.422.1 Detailed Description . . . 1010
12.422.2 Member Function Documentation . . . 1010
12.422.2.1 set_pdi() . . . 1010
12.422.2.2 set_PLL_BW() . . . 1010
12.423 Tracking_FLL_PLL_filter Class Reference . . . 1010
12.423.1 Detailed Description . . . 1011
12.424 Tracking_loop_filter Class Reference . . . 1011
12.424.1 Detailed Description . . . 1011
12.424.2 Constructor & Destructor Documentation . . . 1011
12.424.2.1 Tracking_loop_filter() . . . 1011
12.424.3 Member Function Documentation . . . 1011
12.424.3.1 operator=() . . . 1011
12.425 TrackingCmd Class Reference . . . 1012
12.425.1 Detailed Description . . . 1012
12.425.2 Member Data Documentation . . . 1012
12.425.2.1 carrier_freq_hz . . . 1012
12.425.2.2 carrier_freq_rate_hz_s . . . 1012
12.425.2.3 code_freq_chips . . . 1012
12.425.2.4 enable_carrier_nco_cmd . . . 1012
12.425.2.5 enable_code_nco_cmd . . . 1012

12.425.2.6 sample_counter . . . 1012
12.426 TrackingInterface Class Reference . . . 1012
12.426.1 Detailed Description . . . 1013
12.426.2 Member Function Documentation . . . 1013
12.426.2.1 set_channel() . . . 1013
12.426.2.2 set_gnss_synchro() . . . 1013
12.426.2.3 start_tracking() . . . 1013
12.426.2.4 stop_tracking() . . . 1014
12.427 trop_t Struct Reference . . . 1014
12.427.1 Detailed Description . . . 1014
12.427.2 Member Data Documentation . . . 1014
12.427.2.1 std . . . 1014
12.427.2.2 time . . . 1014
12.427.2.3 trp . . . 1014
12.428 TwoBitCpxFileSignalSource Class Reference . . . 1014
12.428.1 Detailed Description . . . 1016
12.428.2 Member Function Documentation . . . 1016
12.428.2.1 create_file_source_hook() . . . 1016
12.428.2.2 itemTypeToSize() . . . 1016
12.428.2.3 packetsPerSample() . . . 1017
12.428.2.4 pre_connect_hook() . . . 1017
12.428.2.5 pre_disconnect_hook() . . . 1017
12.428.2.6 source() . . . 1017
12.429 TwoBitPackedFileSignalSource Class Reference . . . 1017
12.429.1 Detailed Description . . . 1019
12.429.2 Member Function Documentation . . . 1019
12.429.2.1 create_file_source_hook() . . . 1019
12.429.2.2 itemTypeToSize() . . . 1019
12.429.2.3 packetsPerSample() . . . 1019
12.429.2.4 pre_connect_hook() . . . 1019
12.429.2.5 pre_disconnect_hook() . . . 1020
12.429.2.6 source() . . . 1020
12.430 UhdSignalSource Class Reference . . . 1020
12.430.1 Detailed Description . . . 1021
12.430.2 Member Function Documentation . . . 1021
12.430.2.1 connect() . . . 1021
12.430.2.2 disconnect() . . . 1021
12.430.2.3 get_left_block() . . . 1021
12.430.2.4 get_right_block() [1/2] . . . 1021
12.430.2.5 get_right_block() [2/2] . . . 1021
12.430.2.6 item_size() . . . 1021
12.431 unpack_2bit_samples Class Reference . . . 1021

12.431.1 Detailed Description . . . 1022
12.432 unpack_byte_2bit_cpx_samples Class Reference . . . 1022
12.432.1 Detailed Description . . . 1022
12.433 unpack_byte_2bit_samples Class Reference . . . 1023
12.433.1 Detailed Description . . . 1023
12.434 unpack_byte_4bit_samples Class Reference . . . 1023
12.434.1 Detailed Description . . . 1023
12.435 unpack_gss6450_samples Class Reference . . . 1024
12.435.1 Detailed Description . . . 1024
12.436 unpack_intspir_1bit_samples Class Reference . . . 1024
12.436.1 Detailed Description . . . 1024
12.437 unpack_ntlab_2bit_samples Class Reference . . . 1025
12.437.1 Detailed Description . . . 1025
12.438 unpack_short_byte_samples Class Reference . . . 1025
12.438.1 Detailed Description . . . 1026
12.439 UnscentedFilter Class Reference . . . 1026
12.439.1 Detailed Description . . . 1026
12.440 url_t Struct Reference . . . 1026
12.440.1 Detailed Description . . . 1026
12.440.2 Member Data Documentation . . . 1026
12.440.2.1 dir . . . 1026
12.440.2.2 path . . . 1026
12.440.2.3 tint . . . 1026
12.440.2.4 type . . . 1027
12.441 v27_decision_t Struct Reference . . . 1027
12.441.1 Detailed Description . . . 1027
12.441.2 Member Data Documentation . . . 1027
12.441.2.1 w . . . 1027
12.442 v27_poly_t Struct Reference . . . 1027
12.442.1 Detailed Description . . . 1027
12.442.2 Member Data Documentation . . . 1027
12.442.2.1 c0 . . . 1027
12.442.2.2 c1 . . . 1027
12.443 v27_t Struct Reference . . . 1027
12.443.1 Detailed Description . . . 1028
12.443.2 Member Data Documentation . . . 1028
12.443.2.1 decisions . . . 1028
12.443.2.2 decisions_count . . . 1028
12.443.2.3 decisions_index . . . 1028
12.443.2.4 metrics1 . . . 1028
12.443.2.5 metrics2 . . . 1028
12.443.2.6 new_metrics . . . 1028

12.443.2.7 old_metrics . . . 1028
12.443.2.8 poly . . . 1028
12.444 Viterbi_Decoder Class Reference . . . 1028
12.444.1 Detailed Description . . . 1029
12.444.2 Constructor & Destructor Documentation . . . 1029
12.444.2.1 Viterbi_Decoder() . . . 1029
12.444.3 Member Function Documentation . . . 1029
12.444.3.1 decode() . . . 1029
12.444.3.2 reset() . . . 1029
12.445 Viterbi_Decoder_Sbas Class Reference . . . 1030
12.445.1 Detailed Description . . . 1030
12.445.2 Member Function Documentation . . . 1030
12.445.2.1 decode_block() . . . 1030
12.446 ZmqSignalSource Class Reference . . . 1030
12.446.1 Detailed Description . . . 1031
12.446.2 Member Function Documentation . . . 1032
12.446.2.1 connect() . . . 1032
12.446.2.2 disconnect() . . . 1032
12.446.2.3 get_right_block() . . . 1032
12.446.2.4 item_size() . . . 1032

13 File Documentation **1033**

13.1 pcps_acquisition_adapter.h File Reference . . . 1033
13.1.1 Detailed Description . . . 1033
13.2 pcps_acquisition_adapter.h . . . 1033
13.3 pcps_acquisition_adapter_custom.h . . . 1035
13.4 pcps_acquisition_adapter_fpga.h File Reference . . . 1037
13.4.1 Detailed Description . . . 1037
13.5 pcps_acquisition_adapter_fpga.h . . . 1037
13.6 acquisition_impl_interface.h File Reference . . . 1038
13.6.1 Detailed Description . . . 1038
13.7 acquisition_impl_interface.h . . . 1039
13.8 galileo_e5a_noncoherent_iq_acquisition_caf_cc.h File Reference . . . 1040
13.8.1 Detailed Description . . . 1040
13.9 galileo_e5a_noncoherent_iq_acquisition_caf_cc.h . . . 1040
13.10 galileo_pcps_8ms_acquisition_cc.h File Reference . . . 1043
13.10.1 Detailed Description . . . 1043
13.11 galileo_pcps_8ms_acquisition_cc.h . . . 1044
13.12 pcps_acquisition.h File Reference . . . 1046
13.12.1 Detailed Description . . . 1046
13.13 pcps_acquisition.h . . . 1047
13.14 pcps_acquisition_fine_doppler_cc.h File Reference . . . 1051

13.14.1 Detailed Description . . . 1051
13.15 pcps_acquisition_fine_doppler_cc.h . . . 1052
13.16 pcps_acquisition_fpga.h File Reference . . . 1054
13.16.1 Detailed Description . . . 1055
13.17 pcps_acquisition_fpga.h . . . 1055
13.18 pcps_assisted_acquisition_cc.h File Reference . . . 1057
13.18.1 Detailed Description . . . 1058
13.19 pcps_assisted_acquisition_cc.h . . . 1058
13.20 pcps_cccwsr_acquisition_cc.h File Reference . . . 1060
13.20.1 Detailed Description . . . 1061
13.21 pcps_cccwsr_acquisition_cc.h . . . 1061
13.22 pcps_opencl_acquisition_cc.h File Reference . . . 1063
13.22.1 Detailed Description . . . 1064
13.22.2 Macro Definition Documentation . . . 1064
13.22.2.1 CL_SILENCE_DEPRECATION . . . 1064
13.23 pcps_opencl_acquisition_cc.h . . . 1065
13.24 pcps_quicksync_acquisition_cc.h File Reference . . . 1067
13.24.1 Detailed Description . . . 1068
13.25 pcps_quicksync_acquisition_cc.h . . . 1068
13.26 pcps_tong_acquisition_cc.h File Reference . . . 1071
13.26.1 Detailed Description . . . 1071
13.27 pcps_tong_acquisition_cc.h . . . 1072
13.28 acq_conf.h File Reference . . . 1074
13.28.1 Detailed Description . . . 1074
13.29 acq_conf.h . . . 1075
13.30 acq_conf_fpga.h File Reference . . . 1076
13.30.1 Detailed Description . . . 1077
13.31 acq_conf_fpga.h . . . 1077
13.32 cuda_pcps_engine.h File Reference . . . 1078
13.32.1 Detailed Description . . . 1078
13.33 cuda_pcps_engine.h . . . 1078
13.34 fpga_acquisition.h File Reference . . . 1080
13.34.1 Detailed Description . . . 1080
13.35 fpga_acquisition.h . . . 1080
13.36 channel.h File Reference . . . 1083
13.36.1 Detailed Description . . . 1084
13.37 channel.h . . . 1084
13.38 channel_fsm.h File Reference . . . 1085
13.38.1 Detailed Description . . . 1086
13.39 channel_fsm.h . . . 1086
13.40 channel_msg_receiver_cc.h File Reference . . . 1087
13.40.1 Detailed Description . . . 1087

13.41 channel_msg_receiver_cc.h . . . 1088
13.42 array_signal_conditioner.h File Reference . . . 1088
13.42.1 Detailed Description . . . 1089
13.43 array_signal_conditioner.h . . . 1089
13.44 signal_conditioner.h File Reference . . . 1090
13.44.1 Detailed Description . . . 1090
13.45 signal_conditioner.h . . . 1090
13.46 byte_to_short.h File Reference . . . 1091
13.46.1 Detailed Description . . . 1092
13.47 byte_to_short.h . . . 1092
13.48 cshort_to_grcomplex.h File Reference . . . 1093
13.48.1 Detailed Description . . . 1093
13.49 cshort_to_grcomplex.h . . . 1093
13.50 ibyte_to_cbyte.h File Reference . . . 1094
13.50.1 Detailed Description . . . 1095
13.51 ibyte_to_cbyte.h . . . 1095
13.52 ibyte_to_complex.h File Reference . . . 1096
13.52.1 Detailed Description . . . 1096
13.53 ibyte_to_complex.h . . . 1096
13.54 ibyte_to_cshort.h File Reference . . . 1097
13.54.1 Detailed Description . . . 1098
13.55 ibyte_to_cshort.h . . . 1098
13.56 ishort_to_complex.h File Reference . . . 1099
13.56.1 Detailed Description . . . 1099
13.57 ishort_to_complex.h . . . 1099
13.58 ishort_to_cshort.h File Reference . . . 1100
13.58.1 Detailed Description . . . 1101
13.59 ishort_to_cshort.h . . . 1101
13.60 cshort_to_gr_complex.h File Reference . . . 1102
13.60.1 Detailed Description . . . 1102
13.61 cshort_to_gr_complex.h . . . 1103
13.62 interleaved_byte_to_complex_byte.h File Reference . . . 1103
13.62.1 Detailed Description . . . 1104
13.63 interleaved_byte_to_complex_byte.h . . . 1104
13.64 interleaved_byte_to_complex_short.h File Reference . . . 1104
13.64.1 Detailed Description . . . 1105
13.65 interleaved_byte_to_complex_short.h . . . 1105
13.66 interleaved_short_to_complex_short.h File Reference . . . 1106
13.66.1 Detailed Description . . . 1106
13.67 interleaved_short_to_complex_short.h . . . 1106
13.68 beamformer_filter.h File Reference . . . 1107
13.68.1 Detailed Description . . . 1107

13.69 beamformer_filter.h . . . 1107
13.70 fir_filter.h File Reference . . . 1108
13.70.1 Detailed Description . . . 1109
13.71 fir_filter.h . . . 1109
13.72 freq_xlating_fir_filter.h File Reference . . . 1111
13.72.1 Detailed Description . . . 1111
13.73 freq_xlating_fir_filter.h . . . 1111
13.74 notch_filter.h File Reference . . . 1113
13.74.1 Detailed Description . . . 1113
13.75 notch_filter.h . . . 1113
13.76 notch_filter_lite.h File Reference . . . 1114
13.76.1 Detailed Description . . . 1114
13.77 notch_filter_lite.h . . . 1114
13.78 pulse_blanking_filter.h File Reference . . . 1115
13.78.1 Detailed Description . . . 1116
13.79 pulse_blanking_filter.h . . . 1116
13.80 beamformer.h File Reference . . . 1117
13.80.1 Detailed Description . . . 1117
13.81 beamformer.h . . . 1118
13.82 notch_cc.h File Reference . . . 1118
13.82.1 Detailed Description . . . 1119
13.83 notch_cc.h . . . 1119
13.84 notch_lite_cc.h File Reference . . . 1120
13.84.1 Detailed Description . . . 1121
13.85 notch_lite_cc.h . . . 1121
13.86 pulse_blanking_cc.h File Reference . . . 1122
13.86.1 Detailed Description . . . 1122
13.87 pulse_blanking_cc.h . . . 1122
13.88 beidou_b1c_signal_replica.h File Reference . . . 1123
13.88.1 Detailed Description . . . 1124
13.89 beidou_b1c_signal_replica.h . . . 1124
13.90 beidou_b1i_signal_replica.h File Reference . . . 1125
13.90.1 Detailed Description . . . 1125
13.91 beidou_b1i_signal_replica.h . . . 1125
13.92 beidou_b2a_signal_replica.h File Reference . . . 1126
13.92.1 Detailed Description . . . 1126
13.93 beidou_b2a_signal_replica.h . . . 1126
13.94 beidou_b3i_signal_replica.h File Reference . . . 1127
13.94.1 Detailed Description . . . 1127
13.95 beidou_b3i_signal_replica.h . . . 1128
13.96 byte_x2_to_complex_byte.h File Reference . . . 1128
13.96.1 Detailed Description . . . 1129

13.97 byte_x2_to_complex_byte.h . . . 1129
13.98 complex_byte_to_float_x2.h File Reference . . . 1130
13.98.1 Detailed Description . . . 1130
13.99 complex_byte_to_float_x2.h . . . 1130
13.100 complex_float_to_complex_byte.h File Reference . . . 1131
13.100.1 Detailed Description . . . 1131
13.101 complex_float_to_complex_byte.h . . . 1131
13.102 conjugate_cc.h File Reference . . . 1132
13.102.1 Detailed Description . . . 1132
13.103 conjugate_cc.h . . . 1133
13.104 conjugate_ic.h File Reference . . . 1133
13.104.1 Detailed Description . . . 1134
13.105 conjugate_ic.h . . . 1134
13.106 conjugate_sc.h File Reference . . . 1135
13.106.1 Detailed Description . . . 1135
13.107 conjugate_sc.h . . . 1135
13.108 cshort_to_float_x2.h File Reference . . . 1136
13.108.1 Detailed Description . . . 1136
13.109 cshort_to_float_x2.h . . . 1137
13.110 dump_logger_helper.h File Reference . . . 1137
13.110.1 Detailed Description . . . 1137
13.110.2 Function Documentation . . . 1138
13.110.2.1 write_value() . . . 1138
13.111 dump_logger_helper.h . . . 1138
13.112 galileo_e1_signal_replica.h File Reference . . . 1138
13.112.1 Detailed Description . . . 1139
13.113 galileo_e1_signal_replica.h . . . 1139
13.114 galileo_e5_signal_replica.h File Reference . . . 1140
13.114.1 Detailed Description . . . 1140
13.115 galileo_e5_signal_replica.h . . . 1141
13.116 galileo_e6_signal_replica.h File Reference . . . 1142
13.116.1 Detailed Description . . . 1142
13.117 galileo_e6_signal_replica.h . . . 1143
13.118 geofunctions.h File Reference . . . 1144
13.118.1 Detailed Description . . . 1145
13.119 geofunctions.h . . . 1145
13.120 glonass_l1_signal_replica.h . . . 1147
13.121 glonass_l2_signal_replica.h File Reference . . . 1148
13.121.1 Detailed Description . . . 1148
13.122 glonass_l2_signal_replica.h . . . 1148
13.123 gnss_circular_deque.h File Reference . . . 1149
13.123.1 Detailed Description . . . 1149

13.124 gnss_circular_deque.h . . . 1150
13.125 gnss_sdr_create_directory.h File Reference . . . 1151
13.125.1 Detailed Description . . . 1152
13.126 gnss_sdr_create_directory.h . . . 1152
13.127 gnss_sdr_fft.h File Reference . . . 1152
13.127.1 Detailed Description . . . 1153
13.127.2 Typedef Documentation . . . 1153
13.127.2.1 gnss_fft_complex_fwd . . . 1153
13.127.2.2 gnss_fft_complex_rev . . . 1153
13.127.2.3 gnss_fft_fwd_unique_ptr . . . 1153
13.127.2.4 gnss_fft_rev_unique_ptr . . . 1153
13.127.3 Function Documentation . . . 1153
13.127.3.1 gnss_fft_fwd_make_unique() . . . 1153
13.127.3.2 gnss_fft_rev_make_unique() . . . 1153
13.128 gnss_sdr_fft.h . . . 1154
13.129 gnss_sdr_filesystem.h File Reference . . . 1154
13.129.1 Detailed Description . . . 1155
13.130 gnss_sdr_filesystem.h . . . 1155
13.131 gnss_sdr_flags.h File Reference . . . 1155
13.131.1 Detailed Description . . . 1157
13.132 gnss_sdr_flags.h . . . 1157
13.133 gnss_sdr_make_unique.h File Reference . . . 1159
13.133.1 Detailed Description . . . 1159
13.134 gnss_sdr_make_unique.h . . . 1159
13.135 gnss_sdr_string_literals.h File Reference . . . 1161
13.135.1 Detailed Description . . . 1161
13.136 gnss_sdr_string_literals.h . . . 1161
13.137 gnss_signal_replica.h File Reference . . . 1161
13.137.1 Detailed Description . . . 1162
13.138 gnss_signal_replica.h . . . 1162
13.139 gnss_time.h File Reference . . . 1163
13.139.1 Detailed Description . . . 1163
13.140 gnss_time.h . . . 1164
13.141 gps_l2c_signal_replica.h File Reference . . . 1164
13.141.1 Detailed Description . . . 1164
13.142 gps_l2c_signal_replica.h . . . 1165
13.143 gps_l5_signal_replica.h File Reference . . . 1165
13.143.1 Detailed Description . . . 1166
13.144 gps_l5_signal_replica.h . . . 1166
13.145 gps_sdr_signal_replica.h File Reference . . . 1167
13.145.1 Detailed Description . . . 1167
13.146 gps_sdr_signal_replica.h . . . 1167

13.147 item_type_helpers.h File Reference . . . 1168
13.147.1 Detailed Description . . . 1169
13.148 item_type_helpers.h . . . 1169
13.149 matlab_writter_helper.h File Reference . . . 1170
13.149.1 Detailed Description . . . 1170
13.149.2 Function Documentation . . . 1170
13.149.2.1 write_matlab_var() [1/2] . . . 1170
13.149.2.2 write_matlab_var() [2/2] . . . 1171
13.150 matlab_writter_helper.h . . . 1171
13.151 clFFT.h File Reference . . . 1172
13.151.1 Detailed Description . . . 1173
13.151.2 Typedef Documentation . . . 1173
13.151.2.1 clFFT_Plan . . . 1173
13.151.3 Enumeration Type Documentation . . . 1173
13.151.3.1 clFFT_DataFormat . . . 1173
13.151.3.2 clFFT_Dimension . . . 1173
13.151.3.3 clFFT_Direction . . . 1173
13.152 clFFT.h . . . 1173
13.153 fft_base_kernels.h File Reference . . . 1174
13.153.1 Detailed Description . . . 1174
13.154 fft_base_kernels.h . . . 1175
13.155 fft_internal.h File Reference . . . 1178
13.155.1 Detailed Description . . . 1178
13.155.2 Enumeration Type Documentation . . . 1178
13.155.2.1 kernel_dir_t . . . 1178
13.156 fft_internal.h . . . 1178
13.157 pass_through.h File Reference . . . 1180
13.157.1 Detailed Description . . . 1180
13.158 pass_through.h . . . 1180
13.159 qzss_signal_replica.h File Reference . . . 1181
13.159.1 Detailed Description . . . 1182
13.160 qzss_signal_replica.h . . . 1182
13.161 rtklib.h File Reference . . . 1183
13.161.1 Detailed Description . . . 1192
13.162 rtklib.h . . . 1193
13.163 rtklib_conversions.h File Reference . . . 1209
13.163.1 Detailed Description . . . 1210
13.164 rtklib_conversions.h . . . 1210
13.165 rtklib_ephemeris.h File Reference . . . 1212
13.165.1 Detailed Description . . . 1213
13.165.2 Variable Documentation . . . 1213
13.165.2.1 BDS_EPH_SELECTION_CNAV1_PREFERRED . . . 1213

13.166 rtklib_ephemeris.h . . . 1213
13.167 rtklib_ionex.h File Reference . . . 1214
13.167.1 Detailed Description . . . 1215
13.167.2 Variable Documentation . . . 1215
13.167.2.1 MIN_EL . . . 1215
13.167.2.2 MIN_HGT . . . 1215
13.167.2.3 VAR_NOTEC . . . 1216
13.168 rtklib_ionex.h . . . 1216
13.169 rtklib_lambda.h File Reference . . . 1217
13.169.1 Detailed Description . . . 1217
13.169.2 Macro Definition Documentation . . . 1217
13.169.2.1 ROUND_LAMBDA . . . 1217
13.169.2.2 SGN_LAMBDA . . . 1218
13.169.2.3 SWAP_LAMBDA . . . 1218
13.169.3 Variable Documentation . . . 1218
13.169.3.1 LOOPMAX . . . 1218
13.170 rtklib_lambda.h . . . 1218
13.171 rtklib_pntpos.h File Reference . . . 1219
13.171.1 Detailed Description . . . 1220
13.171.2 Function Documentation . . . 1220
13.171.2.1 pntpos() . . . 1220
13.171.3 Variable Documentation . . . 1221
13.171.3.1 ERR_ION . . . 1221
13.171.3.2 ERR_TROP . . . 1221
13.171.3.3 MAXITR . . . 1221
13.171.3.4 NX . . . 1221
13.171.4 of estimated parameters: pos (3), GPS clock, GLO/GAL/BDS/QZS-GPS offsets . . . 1221
13.171.4.1 VARERR_MIN_EL . . . 1221
13.172 rtklib_pntpos.h . . . 1221
13.173 rtklib_ppp.h File Reference . . . 1223
13.173.1 Detailed Description . . . 1225
13.173.2 Macro Definition Documentation . . . 1225
13.173.2.1 MIN_PPP . . . 1225
13.173.2.2 ROUND_PPP . . . 1225
13.173.2.3 SWAP_D . . . 1225
13.173.2.4 SWAP_I . . . 1225
13.173.3 Variable Documentation . . . 1226
13.173.3.1 CONST_AMB . . . 1226
13.173.3.2 LOG_PI . . . 1226
13.173.3.3 MIN_ARC_GAP . . . 1226
13.173.3.4 SQRT2 . . . 1226
13.173.3.5 THRES_RES . . . 1226

13.173.3.6 VAR_BIAS . . . 1226
13.173.3.7 VAR_CLK . . . 1226
13.173.3.8 VAR_GRA_PPP . . . 1226
13.173.3.9 VAR_IONO_OFF . . . 1226
13.173.3.10 VAR_POS_PPP . . . 1226
13.173.3.11 VAR_ZTD . . . 1227
13.174 rtklib_ppp.h . . . 1227
13.175 rtklib_preceph.h File Reference . . . 1229
13.175.1 Detailed Description . . . 1229
13.175.2 Variable Documentation . . . 1230
13.175.2.1 EXTERR_CLK . . . 1230
13.175.2.2 EXTERR_EPH . . . 1230
13.175.2.3 MAXDTE . . . 1230
13.175.2.4 NMAX . . . 1230
13.176 rtklib_preceph.h . . . 1230
13.177 rtklib_rtcm.h File Reference . . . 1231
13.177.1 Detailed Description . . . 1231
13.177.2 Macro Definition Documentation . . . 1232
13.177.2.1 RTCM2PREAMB . . . 1232
13.177.2.2 RTCM3PREAMB . . . 1232
13.178 rtklib_rtcm.h . . . 1232
13.179 rtklib_rtcm2.h File Reference . . . 1233
13.179.1 Detailed Description . . . 1233
13.180 rtklib_rtcm2.h . . . 1234
13.181 rtklib_rtcm3.h File Reference . . . 1234
13.181.1 Detailed Description . . . 1236
13.181.2 Variable Documentation . . . 1236
13.181.2.1 CODES_BDS . . . 1236
13.181.2.2 CODES_GAL . . . 1237
13.181.2.3 CODES_GLO . . . 1237
13.181.2.4 CODES_GPS . . . 1237
13.181.2.5 CODES_QZS . . . 1237
13.181.2.6 CODES_SBS . . . 1237
13.181.2.7 PRUNIT_GLO . . . 1237
13.181.2.8 PRUNIT_GPS . . . 1237
13.181.2.9 RANGE_MS . . . 1237
13.181.2.10 SSRUDINT . . . 1238
13.182 rtklib_rtcm3.h . . . 1238
13.183 rtklib_rtkcmn.h File Reference . . . 1240
13.183.1 Detailed Description . . . 1244
13.183.2 Macro Definition Documentation . . . 1244
13.183.2.1 Rx . . . 1244

13.183.2.2 Ry . . . 1244
13.183.2.3 Rz . . . 1245
13.184 rtklib_rtkcmn.h . . . 1245
13.185 rtklib_rtkpos.h File Reference . . . 1248
13.185.1 Detailed Description . . . 1250
13.185.1.1 SPDX-License-Identifier: BSD-2-Clause . . . 1250
13.186 rtklib_rtkpos.h . . . 1250
13.187 rtklib_rtksvr.h File Reference . . . 1252
13.187.1 Detailed Description . . . 1253
13.187.2 Variable Documentation . . . 1253
13.187.2.1 PRCOPT_DEFAULT . . . 1253
13.187.2.2 SOLOPT_DEFAULT . . . 1254
13.188 rtklib_rtksvr.h . . . 1254
13.189 rtklib_sbas.h File Reference . . . 1255
13.189.1 Detailed Description . . . 1256
13.189.2 Variable Documentation . . . 1257
13.189.2.1 IGPBAND1 . . . 1257
13.189.2.2 IGPBAND2 . . . 1257
13.189.2.3 WEEKOFFSET . . . 1257
13.190 rtklib_sbas.h . . . 1257
13.191 rtklib_secure_wipe.h File Reference . . . 1259
13.191.1 Detailed Description . . . 1259
13.192 rtklib_secure_wipe.h . . . 1259
13.193 rtklib_solution.h File Reference . . . 1260
13.193.1 Detailed Description . . . 1261
13.193.1.1 SPDX-License-Identifier: BSD-2-Clause . . . 1261
13.194 rtklib_solution.h . . . 1262
13.195 rtklib_stream.h File Reference . . . 1264
13.195.1 Detailed Description . . . 1266
13.195.2 Macro Definition Documentation . . . 1266
13.195.2.1 FTP_CMD . . . 1266
13.195.2.2 FTP_TIMEOUT . . . 1266
13.195.2.3 MAXCLI . . . 1266
13.195.2.4 MAXSTATMSG . . . 1266
13.195.2.5 NTRIP_AGENT . . . 1266
13.195.2.6 NTRIP_CLI_PORT . . . 1266
13.195.2.7 NTRIP_MAXRSP . . . 1266
13.195.2.8 NTRIP_MAXSTR . . . 1266
13.195.2.9 NTRIP_RSP_ERROR . . . 1267
13.195.2.10 NTRIP_RSP_HTTP . . . 1267
13.195.2.11 NTRIP_RSP_OK_CLI . . . 1267
13.195.2.12 NTRIP_RSP_OK_SVR . . . 1267

13.195.2.13 NTRIP_RSP_SRCTBL . . . 1267
13.195.2.14 NTRIP_RSP_TBLEND . . . 1267
13.195.2.15 NTRIP_SVR_PORT . . . 1267
13.195.2.16 NTRIP_VERSION_1 . . . 1267
13.195.2.17 NTRIP_VERSION_2 . . . 1267
13.195.2.18 SERIBUFFSIZE . . . 1267
13.195.2.19 TIMETAGH_LEN . . . 1267
13.195.2.20 TINTACT . . . 1267
13.195.2.21 VER_RTKLIB . . . 1268
13.196 rtklib_stream.h . . . 1268
13.197 rtklib_tides.h File Reference . . . 1270
13.197.1 Detailed Description . . . 1271
13.197.1.1 Conventions (2010), 2010 . . . 1271
13.197.2 Variable Documentation . . . 1271
13.197.2.1 GME . . . 1271
13.197.2.2 GMM . . . 1271
13.197.2.3 GMS . . . 1271
13.198 rtklib_tides.h . . . 1271
13.199 rtklib_tls.h File Reference . . . 1272
13.199.1 Detailed Description . . . 1273
13.200 rtklib_tls.h . . . 1273
13.201 sbas_signal_replica.h File Reference . . . 1274
13.201.1 Detailed Description . . . 1274
13.202 sbas_signal_replica.h . . . 1274
13.203 sensor_data_aggregator.h File Reference . . . 1275
13.203.1 Detailed Description . . . 1275
13.204 sensor_data_aggregator.h . . . 1275
13.205 sensor_data_file.h File Reference . . . 1276
13.205.1 Detailed Description . . . 1277
13.206 sensor_data_file.h . . . 1277
13.207 sensor_data_resampler.h File Reference . . . 1278
13.207.1 Detailed Description . . . 1278
13.208 sensor_data_resampler.h . . . 1278
13.209 sensor_data_source.h File Reference . . . 1279
13.209.1 Detailed Description . . . 1279
13.210 sensor_data_source.h . . . 1279
13.211 sensor_data_source_configuration.h File Reference . . . 1280
13.211.1 Detailed Description . . . 1280
13.212 sensor_data_source_configuration.h . . . 1280
13.213 sensor_data_type.h File Reference . . . 1282
13.213.1 Detailed Description . . . 1282
13.214 sensor_data_type.h . . . 1282

13.215 sensor_identifier.h File Reference . . . 1283
13.215.1 Detailed Description . . . 1283
13.216 sensor_identifier.h . . . 1283
13.217 short_x2_to_cshort.h File Reference . . . 1284
13.217.1 Detailed Description . . . 1284
13.218 short_x2_to_cshort.h . . . 1285
13.219 trackingcmd.h File Reference . . . 1285
13.219.1 Detailed Description . . . 1285
13.220 trackingcmd.h . . . 1286
13.221 hybrid_observables.h File Reference . . . 1286
13.221.1 Detailed Description . . . 1287
13.222 hybrid_observables.h . . . 1287
13.223 hybrid_observables_gs.h File Reference . . . 1288
13.223.1 Detailed Description . . . 1288
13.224 hybrid_observables_gs.h . . . 1289
13.225 obs_conf.h File Reference . . . 1292
13.225.1 Detailed Description . . . 1292
13.226 obs_conf.h . . . 1292
13.227 rtklib_pvt.h File Reference . . . 1293
13.227.1 Detailed Description . . . 1293
13.228 rtklib_pvt.h . . . 1293
13.229 rtklib_pvt_gs.h File Reference . . . 1297
13.229.1 Detailed Description . . . 1298
13.230 rtklib_pvt_gs.h . . . 1298
13.231 an_packet_printer.h File Reference . . . 1303
13.231.1 Detailed Description . . . 1303
13.232 an_packet_printer.h . . . 1303
13.233 galileo_ephemeris_store.h File Reference . . . 1305
13.233.1 Detailed Description . . . 1305
13.234 galileo_ephemeris_store.h . . . 1306
13.235 geohash.h File Reference . . . 1308
13.235.1 Detailed Description . . . 1308
13.236 geohash.h . . . 1308
13.237 geojson_printer.h File Reference . . . 1309
13.237.1 Detailed Description . . . 1309
13.238 geojson_printer.h . . . 1309
13.239 gpx_printer.h File Reference . . . 1310
13.239.1 Detailed Description . . . 1310
13.240 gpx_printer.h . . . 1310
13.241 has_simple_printer.h File Reference . . . 1311
13.241.1 Detailed Description . . . 1311
13.242 has_simple_printer.h . . . 1312

13.243 kml_printer.h File Reference . . . 1313
13.243.1 Detailed Description . . . 1313
13.244 kml_printer.h . . . 1313
13.245 monitor_ephemeris_udp_sink.h File Reference . . . 1314
13.245.1 Detailed Description . . . 1314
13.246 monitor_ephemeris_udp_sink.h . . . 1314
13.247 monitor_pvt.h File Reference . . . 1315
13.247.1 Detailed Description . . . 1315
13.247.1.1 Álvaro Cebrián Juan, 2019. acebrianjuan(at)gmail.com . . . 1316
13.248 monitor_pvt.h . . . 1316
13.249 monitor_pvt_udp_sink.h File Reference . . . 1318
13.249.1 Detailed Description . . . 1319
13.250 monitor_pvt_udp_sink.h . . . 1319
13.251 nmea_printer.h File Reference . . . 1320
13.251.1 Detailed Description . . . 1320
13.252 nmea_printer.h . . . 1320
13.253 ntrip_rtcm_client.h File Reference . . . 1321
13.253.1 Detailed Description . . . 1322
13.254 ntrip_rtcm_client.h . . . 1322
13.255 ntrip_rtk_signals.h File Reference . . . 1325
13.255.1 Detailed Description . . . 1326
13.256 ntrip_rtk_signals.h . . . 1326
13.257 pvt_conf.h File Reference . . . 1329
13.257.1 Detailed Description . . . 1329
13.258 pvt_conf.h . . . 1329
13.259 pvt_kf.h File Reference . . . 1331
13.259.1 Detailed Description . . . 1331
13.260 pvt_kf.h . . . 1331
13.261 pvt_solution.h File Reference . . . 1332
13.261.1 Detailed Description . . . 1332
13.262 pvt_solution.h . . . 1333
13.263 rinex_printer.h File Reference . . . 1334
13.263.1 Detailed Description . . . 1334
13.264 rinex_printer.h . . . 1335
13.265 rtcm.h File Reference . . . 1339
13.265.1 Detailed Description . . . 1339
13.266 rtcm.h . . . 1340
13.267 rtcm_printer.h File Reference . . . 1361
13.267.1 Detailed Description . . . 1361
13.268 rtcm_printer.h . . . 1361
13.269 rtklib_solver.h File Reference . . . 1364
13.269.1 Detailed Description . . . 1365

13.270 rtklib_solver.h . . . 1365
13.271 sbas_rtklib_corrections.h File Reference . . . 1370
13.271.1 Detailed Description . . . 1370
13.272 sbas_rtklib_corrections.h . . . 1370
13.273 secure_string.h File Reference . . . 1371
13.273.1 Detailed Description . . . 1371
13.274 secure_string.h . . . 1371
13.275 serdes_galileo_eph.h File Reference . . . 1373
13.275.1 Detailed Description . . . 1373
13.276 serdes_galileo_eph.h . . . 1373
13.277 serdes_gps_eph.h File Reference . . . 1376
13.277.1 Detailed Description . . . 1376
13.278 serdes_gps_eph.h . . . 1376
13.279 serdes_monitor_pvt.h File Reference . . . 1378
13.279.1 Detailed Description . . . 1379
13.280 serdes_monitor_pvt.h . . . 1379
13.281 signal_enabled_flags.h File Reference . . . 1381
13.281.1 Detailed Description . . . 1382
13.282 signal_enabled_flags.h . . . 1382
13.283 direct_resampler_conditioner.h File Reference . . . 1383
13.283.1 Detailed Description . . . 1383
13.284 direct_resampler_conditioner.h . . . 1383
13.285 mmse_resampler_conditioner.h File Reference . . . 1384
13.285.1 Detailed Description . . . 1384
13.286 mmse_resampler_conditioner.h . . . 1385
13.287 direct_resampler_conditioner_cb.h File Reference . . . 1386
13.287.1 Detailed Description . . . 1386
13.288 direct_resampler_conditioner_cb.h . . . 1386
13.289 direct_resampler_conditioner_cc.h File Reference . . . 1388
13.289.1 Detailed Description . . . 1388
13.290 direct_resampler_conditioner_cc.h . . . 1388
13.291 direct_resampler_conditioner_cs.h File Reference . . . 1389
13.291.1 Detailed Description . . . 1390
13.292 direct_resampler_conditioner_cs.h . . . 1390
13.293 signal_generator.h File Reference . . . 1391
13.293.1 Detailed Description . . . 1391
13.294 signal_generator.h . . . 1391
13.295 signal_generator_c.h File Reference . . . 1392
13.295.1 Detailed Description . . . 1393
13.295.2 Typedef Documentation . . . 1393
13.295.2.1 signal_generator_c_sptr . . . 1393
13.295.3 Function Documentation . . . 1393

13.295.3.1 signal_make_generator_c() . . . 1393
13.296 signal_generator_c.h . . . 1394
13.297 ad936x_custom_signal_source.h File Reference . . . 1395
13.297.1 Detailed Description . . . 1396
13.298 ad936x_custom_signal_source.h . . . 1396
13.299 adrv9361_z7035_signal_source_fpga.h File Reference . . . 1398
13.299.1 Detailed Description . . . 1398
13.300 adrv9361_z7035_signal_source_fpga.h . . . 1398
13.301 bladerf_signal_source.h File Reference . . . 1400
13.301.1 Detailed Description . . . 1400
13.302 bladerf_signal_source.h . . . 1400
13.303 custom_udp_signal_source.h File Reference . . . 1401
13.303.1 Detailed Description . . . 1402
13.303.1.1 autotoc_md299 . . . 1402
13.304 custom_udp_signal_source.h . . . 1402
13.305 dma_signal_source_fpga.h File Reference . . . 1403
13.305.1 Detailed Description . . . 1404
13.306 dma_signal_source_fpga.h . . . 1404
13.307 evk1029_signal_source.h File Reference . . . 1405
13.307.1 Detailed Description . . . 1406
13.308 evk1029_signal_source.h . . . 1406
13.309 fifo_signal_source.h File Reference . . . 1407
13.309.1 Detailed Description . . . 1407
13.310 fifo_signal_source.h . . . 1407
13.311 file_signal_source.h File Reference . . . 1408
13.311.1 Detailed Description . . . 1409
13.312 file_signal_source.h . . . 1409
13.313 file_source_base.h File Reference . . . 1410
13.313.1 Detailed Description . . . 1410
13.314 file_source_base.h . . . 1410
13.315 file_timestamp_signal_source.h File Reference . . . 1413
13.315.1 Detailed Description . . . 1413
13.316 file_timestamp_signal_source.h . . . 1413
13.317 flexiband_signal_source.h File Reference . . . 1414
13.317.1 Detailed Description . . . 1414
13.318 flexiband_signal_source.h . . . 1414
13.319 fmcomms2_signal_source.h File Reference . . . 1416
13.319.1 Detailed Description . . . 1416
13.319.1.1 This class represent a fmcomms2 signal source. It use the gr_iio block . . . 1416
13.320 fmcomms2_signal_source.h . . . 1416
13.321 fmcomms5_signal_source_fpga.h File Reference . . . 1418
13.321.1 Detailed Description . . . 1418

13.322 fmcomms5_signal_source_fpga.h . . . 1418
13.323 four_bit_cpx_file_signal_source.h File Reference . . . 1420
13.323.1 Detailed Description . . . 1420
13.324 four_bit_cpx_file_signal_source.h . . . 1420
13.325 gen_signal_source.h File Reference . . . 1421
13.325.1 Detailed Description . . . 1422
13.326 gen_signal_source.h . . . 1422
13.327 gnmax_signal_source.h File Reference . . . 1423
13.327.1 Detailed Description . . . 1423
13.328 gnmax_signal_source.h . . . 1423
13.329 gss6450_file_signal_source.h File Reference . . . 1424
13.329.1 Detailed Description . . . 1425
13.330 gss6450_file_signal_source.h . . . 1425
13.331 ion_gsms_signal_source.h File Reference . . . 1427
13.331.1 Detailed Description . . . 1427
13.332 ion_gsms_signal_source.h . . . 1427
13.333 labsat_signal_source.h File Reference . . . 1429
13.333.1 Detailed Description . . . 1430
13.334 labsat_signal_source.h . . . 1430
13.335 limesdr_signal_source.h . . . 1431
13.336 max2771_evkit_signal_source_fpga.h File Reference . . . 1432
13.336.1 Detailed Description . . . 1432
13.337 max2771_evkit_signal_source_fpga.h . . . 1433
13.338 multichannel_file_signal_source.h File Reference . . . 1435
13.338.1 Detailed Description . . . 1435
13.339 multichannel_file_signal_source.h . . . 1435
13.340 nsr_file_signal_source.h File Reference . . . 1437
13.340.1 Detailed Description . . . 1437
13.341 nsr_file_signal_source.h . . . 1437
13.342 ntlab_file_signal_source.h File Reference . . . 1438
13.342.1 Detailed Description . . . 1439
13.343 ntlab_file_signal_source.h . . . 1439
13.344 osmosdr_signal_source.h File Reference . . . 1440
13.344.1 Detailed Description . . . 1440
13.345 osmosdr_signal_source.h . . . 1440
13.346 plutosdr_signal_source.h File Reference . . . 1441
13.346.1 Detailed Description . . . 1442
13.347 plutosdr_signal_source.h . . . 1442
13.348 pocket_sdr_signal_source.h File Reference . . . 1443
13.348.1 Detailed Description . . . 1444
13.349 pocket_sdr_signal_source.h . . . 1444
13.350 raw_array_signal_source.h File Reference . . . 1445

13.350.1 Detailed Description . . . 1445
13.351 raw_array_signal_source.h . . . 1445
13.352 rtl_tcp_signal_source.h File Reference . . . 1446
13.352.1 Detailed Description . . . 1447
13.353 rtl_tcp_signal_source.h . . . 1447
13.354 signal_source_base.h File Reference . . . 1448
13.354.1 Detailed Description . . . 1448
13.355 signal_source_base.h . . . 1448
13.356 spir_file_signal_source.h File Reference . . . 1449
13.356.1 Detailed Description . . . 1449
13.357 spir_file_signal_source.h . . . 1450
13.358 two_bit_cpx_file_signal_source.h File Reference . . . 1450
13.358.1 Detailed Description . . . 1451
13.359 two_bit_cpx_file_signal_source.h . . . 1451
13.360 two_bit_packed_file_signal_source.h File Reference . . . 1452
13.360.1 Detailed Description . . . 1452
13.361 two_bit_packed_file_signal_source.h . . . 1452
13.362 uhd_signal_source.h File Reference . . . 1454
13.362.1 Detailed Description . . . 1454
13.363 uhd_signal_source.h . . . 1454
13.364 zmq_signal_source.h File Reference . . . 1455
13.364.1 Detailed Description . . . 1456
13.365 zmq_signal_source.h . . . 1456
13.366 ad936x_iio_source.h File Reference . . . 1457
13.366.1 Detailed Description . . . 1457
13.367 ad936x_iio_source.h . . . 1458
13.368 bladerf_source.h File Reference . . . 1460
13.368.1 Detailed Description . . . 1460
13.369 bladerf_source.h . . . 1460
13.370 evk1029_source.h File Reference . . . 1462
13.370.1 Detailed Description . . . 1462
13.371 evk1029_source.h . . . 1463
13.372 fifo_reader.h File Reference . . . 1464
13.372.1 Detailed Description . . . 1464
13.373 fifo_reader.h . . . 1464
13.374 gr_complex_ip_packet_source.h File Reference . . . 1466
13.374.1 Detailed Description . . . 1466
13.375 gr_complex_ip_packet_source.h . . . 1466
13.376 ion_gsms.h File Reference . . . 1467
13.376.1 Detailed Description . . . 1468
13.377 ion_gsms.h . . . 1468
13.378 labsat23_source.h File Reference . . . 1469

13.378.1 Detailed Description . . . 1470
13.379 labsat23_source.h . . . 1470
13.380 rtl_tcp_signal_source_c.h File Reference . . . 1472
13.380.1 Detailed Description . . . 1473
13.381 rtl_tcp_signal_source_c.h . . . 1473
13.382 unpack_2bit_samples.h File Reference . . . 1475
13.382.1 Detailed Description . . . 1475
13.383 unpack_2bit_samples.h . . . 1475
13.384 unpack_byte_2bit_cpx_samples.h File Reference . . . 1477
13.384.1 Detailed Description . . . 1477
13.385 unpack_byte_2bit_cpx_samples.h . . . 1477
13.386 unpack_byte_2bit_samples.h File Reference . . . 1478
13.386.1 Detailed Description . . . 1479
13.387 unpack_byte_2bit_samples.h . . . 1479
13.388 unpack_byte_4bit_samples.h File Reference . . . 1480
13.388.1 Detailed Description . . . 1480
13.389 unpack_byte_4bit_samples.h . . . 1480
13.390 unpack_gss6450_samples.h File Reference . . . 1481
13.390.1 Detailed Description . . . 1481
13.391 unpack_gss6450_samples.h . . . 1481
13.392 unpack_intspir_1bit_samples.h File Reference . . . 1482
13.392.1 Detailed Description . . . 1483
13.393 unpack_intspir_1bit_samples.h . . . 1483
13.394 unpack_ntlab_2bit_samples.h File Reference . . . 1484
13.394.1 Detailed Description . . . 1484
13.395 unpack_ntlab_2bit_samples.h . . . 1484
13.396 unpack_short_byte_samples.h . . . 1485
13.397 ad9361_manager.h File Reference . . . 1486
13.397.1 Detailed Description . . . 1487
13.398 ad9361_manager.h . . . 1487
13.399 ad936x_iio_custom.h File Reference . . . 1489
13.399.1 Detailed Description . . . 1489
13.399.1.1 autotoc_md397 . . . 1489
13.400 ad936x_iio_custom.h . . . 1489
13.401 ad936x_iio_samples.h File Reference . . . 1491
13.401.1 Detailed Description . . . 1491
13.401.2 Macro Definition Documentation . . . 1492
13.401.2.1 IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES . . . 1492
13.401.2.2 IIO_INPUTRAMFIFOSIZE . . . 1492
13.402 ad936x_iio_samples.h . . . 1492
13.403 fpga_buffer_monitor.h File Reference . . . 1492
13.403.1 Detailed Description . . . 1493

13.404 fpga_buffer_monitor.h . . . 1493
13.405 fpga_dma-proxy.h File Reference . . . 1494
13.405.1 Detailed Description . . . 1495
13.405.2 Enumeration Type Documentation . . . 1495
13.405.2.1 proxy_status . . . 1495
13.405.3 Variable Documentation . . . 1495
13.405.3.1 buffer . . . 1495
13.405.3.2 length . . . 1495
13.406 fpga_dma-proxy.h . . . 1495
13.407 fpga_dynamic_bit_selection.h File Reference . . . 1496
13.407.1 Detailed Description . . . 1497
13.408 fpga_dynamic_bit_selection.h . . . 1497
13.409 fpga_spidev.h File Reference . . . 1498
13.409.1 Detailed Description . . . 1498
13.410 fpga_spidev.h . . . 1498
13.411 fpga_switch.h File Reference . . . 1499
13.411.1 Detailed Description . . . 1499
13.412 fpga_switch.h . . . 1500
13.413 gnss_sdr_timestamp.h File Reference . . . 1501
13.413.1 Detailed Description . . . 1501
13.414 gnss_sdr_timestamp.h . . . 1501
13.415 gnss_sdr_valve.h File Reference . . . 1502
13.415.1 Detailed Description . . . 1503
13.416 gnss_sdr_valve.h . . . 1503
13.417 ion_gsms_chunk_data.h File Reference . . . 1504
13.417.1 Detailed Description . . . 1505
13.417.2 Function Documentation . . . 1505
13.417.2.1 bits_to_item_size() . . . 1505
13.417.2.2 with_word_type() . . . 1505
13.418 ion_gsms_chunk_data.h . . . 1505
13.419 ion_gsms_chunk_unpacking_ctx.h File Reference . . . 1508
13.419.1 Detailed Description . . . 1508
13.420 ion_gsms_chunk_unpacking_ctx.h . . . 1508
13.421 ion_gsms_stream_encodings.h File Reference . . . 1509
13.421.1 Detailed Description . . . 1510
13.421.2 Typedef Documentation . . . 1510
13.421.2.1 StreamEncoding . . . 1510
13.421.3 Function Documentation . . . 1510
13.421.3.1 encoding_from_string() . . . 1510
13.421.4 Variable Documentation . . . 1511
13.421.4.1 five_bit_look_up . . . 1511
13.421.4.2 four_bit_look_up . . . 1511

13.421.4.3 FP . . . 1511
13.421.4.4 MS . . . 1511
13.421.4.5 MSA . . . 1511
13.421.4.6 OB . . . 1511
13.421.4.7 OBA . . . 1512
13.421.4.8 OG . . . 1512
13.421.4.9 OGA . . . 1512
13.421.4.10 SIGN . . . 1512
13.421.4.11 SM . . . 1512
13.421.4.12 SMA . . . 1512
13.421.4.13 TC . . . 1512
13.421.4.14 TCA . . . 1512
13.421.4.15 three_bit_look_up . . . 1512
13.421.4.16 two_bit_look_up . . . 1513
13.421.4.17 UNKNOWN . . . 1513
13.422 ion_gsms_stream_encodings.h . . . 1513
13.423 pps_samplestamp.h File Reference . . . 1515
13.423.1 Detailed Description . . . 1515
13.423.1.1 autotoc_md421 . . . 1515
13.424 pps_samplestamp.h . . . 1515
13.425 ppstcprx.h File Reference . . . 1516
13.425.1 Detailed Description . . . 1516
13.425.1.1 autotoc_md423 . . . 1516
13.426 ppstcprx.h . . . 1516
13.427 rtl_tcp_commands.h File Reference . . . 1517
13.427.1 Detailed Description . . . 1517
13.427.1.1 https://git.osmocom.org/rtl-sdr . . . 1518
13.428 rtl_tcp_commands.h . . . 1518
13.429 rtl_tcp_dongle_info.h File Reference . . . 1518
13.429.1 Detailed Description . . . 1519
13.429.1.1 https://git.osmocom.org/rtl-sdr . . . 1519
13.430 rtl_tcp_dongle_info.h . . . 1519
13.431 telemetry_decoder_adapter.h File Reference . . . 1520
13.431.1 Detailed Description . . . 1520
13.432 telemetry_decoder_adapter.h . . . 1520
13.433 beidou_b1c_telemetry_decoder_gs.h File Reference . . . 1521
13.433.1 Detailed Description . . . 1522
13.433.2 Typedef Documentation . . . 1522
13.433.2.1 beidou_b1c_telemetry_decoder_gs_sptr . . . 1522
13.434 beidou_b1c_telemetry_decoder_gs.h . . . 1522
13.435 beidou_b2a_telemetry_decoder_gs.h File Reference . . . 1523
13.435.1 Detailed Description . . . 1523

13.435.2 Typedef Documentation . . . 1523
13.435.2.1 beidou_b2a_telemetry_decoder_gs_sptr . . . 1523
13.436 beidou_b2a_telemetry_decoder_gs.h . . . 1524
13.437 beidou_dnav_telemetry_decoder_gs.h File Reference . . . 1525
13.437.1 Detailed Description . . . 1525
13.438 beidou_dnav_telemetry_decoder_gs.h . . . 1525
13.439 galileo_telemetry_decoder_gs.h File Reference . . . 1527
13.439.1 Detailed Description . . . 1527
13.440 galileo_telemetry_decoder_gs.h . . . 1528
13.441 glonass_gnav_telemetry_decoder_gs.h File Reference . . . 1530
13.441.1 Detailed Description . . . 1530
13.442 glonass_gnav_telemetry_decoder_gs.h . . . 1530
13.443 gps_l1_ca_telemetry_decoder_gs.h File Reference . . . 1532
13.443.1 Detailed Description . . . 1532
13.444 gps_l1_ca_telemetry_decoder_gs.h . . . 1533
13.445 gps_l2c_telemetry_decoder_gs.h File Reference . . . 1534
13.445.1 Detailed Description . . . 1535
13.446 gps_l2c_telemetry_decoder_gs.h . . . 1535
13.447 gps_l5_telemetry_decoder_gs.h File Reference . . . 1536
13.447.1 Detailed Description . . . 1536
13.447.1.1 autotoc_md445 . . . 1537
13.448 gps_l5_telemetry_decoder_gs.h . . . 1537
13.449 sbas_l1_telemetry_decoder_gs.h File Reference . . . 1538
13.449.1 Detailed Description . . . 1538
13.450 sbas_l1_telemetry_decoder_gs.h . . . 1539
13.451 telemetry_impl_interface.h File Reference . . . 1541
13.451.1 Detailed Description . . . 1542
13.452 telemetry_impl_interface.h . . . 1542
13.453 galileo_tow_utils.h File Reference . . . 1543
13.453.1 Detailed Description . . . 1543
13.453.2 Function Documentation . . . 1544
13.453.2.1 add_ms() . . . 1544
13.453.2.2 floor_div() . . . 1544
13.453.2.3 floor_to_second_ms() . . . 1544
13.453.2.4 fnav_current_symbol_delay_ms() . . . 1544
13.453.2.5 inav_current_symbol_delay_ms() . . . 1544
13.453.2.6 sample_counter_delta() . . . 1544
13.453.2.7 seconds_to_ms() . . . 1544
13.453.2.8 week_after_delta() . . . 1544
13.453.2.9 wrap_ms() . . . 1545
13.453.3 Variable Documentation . . . 1545
13.453.3.1 WEEK_MS . . . 1545

13.454 galileo_tow_utils.h . . . 1545
13.455 bits.h File Reference . . . 1545
13.455.1 Detailed Description . . . 1546
13.456 bits.h . . . 1546
13.457 cnav_msg.h File Reference . . . 1547
13.457.1 Detailed Description . . . 1547
13.458 cnav_msg.h . . . 1548
13.459 edc.h File Reference . . . 1549
13.459.1 Detailed Description . . . 1549
13.460 edc.h . . . 1549
13.461 fec.h File Reference . . . 1550
13.461.1 Detailed Description . . . 1550
13.461.2 Macro Definition Documentation . . . 1551
13.461.2.1 V27POLYA . . . 1551
13.461.2.2 V27POLYB . . . 1551
13.462 fec.h . . . 1551
13.463 swift_common.h File Reference . . . 1552
13.463.1 Detailed Description . . . 1552
13.464 swift_common.h . . . 1552
13.465 tlm_conf.h File Reference . . . 1553
13.465.1 Detailed Description . . . 1553
13.466 tlm_conf.h . . . 1553
13.467 tlm_crc_stats.h File Reference . . . 1554
13.467.1 Detailed Description . . . 1554
13.468 tlm_crc_stats.h . . . 1554
13.469 tlm_utils.h File Reference . . . 1555
13.469.1 Detailed Description . . . 1555
13.470 tlm_utils.h . . . 1555
13.471 tow_utils.h File Reference . . . 1556
13.471.1 Detailed Description . . . 1557
13.472 tow_utils.h . . . 1557
13.473 viterbi_decoder.h File Reference . . . 1559
13.473.1 Detailed Description . . . 1559
13.474 viterbi_decoder.h . . . 1559
13.475 viterbi_decoder_sbas.h File Reference . . . 1561
13.475.1 Detailed Description . . . 1561
13.476 viterbi_decoder_sbas.h . . . 1561
13.477 dll_pll_tracking_adapter.h File Reference . . . 1563
13.477.1 Detailed Description . . . 1563
13.478 dll_pll_tracking_adapter.h . . . 1563
13.479 dll_pll_tracking_adapter_fpga.h File Reference . . . 1565
13.479.1 Detailed Description . . . 1565

13.480 dll_pll_tracking_adapter_fpga.h . . . 1565
13.481 galileo_e1_tcp_connector_tracking.h File Reference . . . 1566
13.481.1 Detailed Description . . . 1566
13.482 galileo_e1_tcp_connector_tracking.h . . . 1567
13.483 gps_l1_ca_dll_pll_tracking_gpu.h File Reference . . . 1568
13.483.1 Detailed Description . . . 1568
13.484 gps_l1_ca_dll_pll_tracking_gpu.h . . . 1568
13.485 gps_l1_ca_gaussian_tracking.h File Reference . . . 1570
13.485.1 Detailed Description . . . 1570
13.486 gps_l1_ca_gaussian_tracking.h . . . 1570
13.487 gps_l1_ca_kf_tracking.h File Reference . . . 1571
13.487.1 Detailed Description . . . 1572
13.488 gps_l1_ca_kf_tracking.h . . . 1572
13.489 gps_l1_ca_tcp_connector_tracking.h File Reference . . . 1573
13.489.1 Detailed Description . . . 1573
13.490 gps_l1_ca_tcp_connector_tracking.h . . . 1574
13.491 dll_pll_veml_tracking.h File Reference . . . 1575
13.491.1 Detailed Description . . . 1575
13.492 dll_pll_veml_tracking.h . . . 1576
13.493 dll_pll_veml_tracking_fpga.h File Reference . . . 1579
13.493.1 Detailed Description . . . 1579
13.494 dll_pll_veml_tracking_fpga.h . . . 1580
13.495 galileo_e1_tcp_connector_tracking_cc.h File Reference . . . 1583
13.495.1 Detailed Description . . . 1583
13.496 galileo_e1_tcp_connector_tracking_cc.h . . . 1584
13.497 gps_l1_ca_dll_pll_tracking_gpu_cc.h File Reference . . . 1586
13.497.1 Detailed Description . . . 1586
13.498 gps_l1_ca_dll_pll_tracking_gpu_cc.h . . . 1586
13.499 gps_l1_ca_gaussian_tracking_cc.h File Reference . . . 1589
13.499.1 Detailed Description . . . 1589
13.500 gps_l1_ca_gaussian_tracking_cc.h . . . 1589
13.501 gps_l1_ca_tcp_connector_tracking_cc.h File Reference . . . 1592
13.501.1 Detailed Description . . . 1593
13.502 gps_l1_ca_tcp_connector_tracking_cc.h . . . 1593
13.503 kf_tracking.h . . . 1595
13.504 bayesian_estimation.h File Reference . . . 1597
13.504.1 Detailed Description . . . 1598
13.504.1.1 . . . 1598
13.505 bayesian_estimation.h . . . 1598
13.506 bit_synchronizer.h File Reference . . . 1599
13.506.1 Detailed Description . . . 1599
13.507 bit_synchronizer.h . . . 1600

13.508 cpu_multicorrelator.h File Reference . . . 1604
13.508.1 Detailed Description . . . 1604
13.509 cpu_multicorrelator.h . . . 1604
13.510 cpu_multicorrelator_16sc.h File Reference . . . 1605
13.510.1 Detailed Description . . . 1605
13.511 cpu_multicorrelator_16sc.h . . . 1606
13.512 cpu_multicorrelator_real_codes.h File Reference . . . 1606
13.512.1 Detailed Description . . . 1607
13.513 cpu_multicorrelator_real_codes.h . . . 1607
13.514 cuda_multicorrelator.h File Reference . . . 1608
13.514.1 Detailed Description . . . 1608
13.514.2 Macro Definition Documentation . . . 1608
13.514.2.1 CUDA_CALLABLE_MEMBER_DEVICE . . . 1608
13.514.2.2 CUDA_CALLABLE_MEMBER_GLOBAL . . . 1608
13.515 cuda_multicorrelator.h . . . 1609
13.516 dll_pll_conf.h File Reference . . . 1611
13.516.1 Detailed Description . . . 1611
13.517 dll_pll_conf.h . . . 1611
13.518 dll_pll_conf_fpga.h File Reference . . . 1612
13.518.1 Detailed Description . . . 1613
13.519 dll_pll_conf_fpga.h . . . 1613
13.520 exponential_smoother.h File Reference . . . 1614
13.520.1 Detailed Description . . . 1614
13.521 exponential_smoother.h . . . 1615
13.522 fpga_multicorrelator.h File Reference . . . 1615
13.522.1 Detailed Description . . . 1616
13.523 fpga_multicorrelator.h . . . 1616
13.524 kf_conf.h . . . 1619
13.525 lock_detectors.h File Reference . . . 1620
13.525.1 Detailed Description . . . 1621
13.526 lock_detectors.h . . . 1621
13.527 nonlinear_tracking.h File Reference . . . 1622
13.527.1 Detailed Description . . . 1623
13.527.1.1 . . . 1623
13.528 nonlinear_tracking.h . . . 1623
13.529 tcp_communication.h File Reference . . . 1624
13.529.1 Detailed Description . . . 1625
13.530 tcp_communication.h . . . 1625
13.531 tcp_packet_data.h File Reference . . . 1626
13.531.1 Detailed Description . . . 1626
13.532 tcp_packet_data.h . . . 1626
13.533 tracking_2nd_DLL_filter.h File Reference . . . 1627

13.533.1 Detailed Description . . . 1627
13.534 tracking_2nd_DLL_filter.h . . . 1627
13.535 tracking_2nd_PLL_filter.h File Reference . . . 1628
13.535.1 Detailed Description . . . 1628
13.536 tracking_2nd_PLL_filter.h . . . 1628
13.537 tracking_discriminators.h File Reference . . . 1629
13.537.1 Detailed Description . . . 1630
13.538 tracking_discriminators.h . . . 1630
13.539 tracking_FLL_PLL_filter.h File Reference . . . 1633
13.539.1 Detailed Description . . . 1633
13.540 tracking_FLL_PLL_filter.h . . . 1633
13.541 tracking_loop_filter.h File Reference . . . 1634
13.541.1 Detailed Description . . . 1634
13.542 tracking_loop_filter.h . . . 1634
13.543 acquisition_interface.h File Reference . . . 1635
13.543.1 Detailed Description . . . 1635
13.544 acquisition_interface.h . . . 1636
13.545 channel_interface.h File Reference . . . 1637
13.545.1 Detailed Description . . . 1637
13.546 channel_interface.h . . . 1637
13.547 configuration_interface.h File Reference . . . 1638
13.547.1 Detailed Description . . . 1638
13.548 configuration_interface.h . . . 1638
13.549 gnss_block_interface.h File Reference . . . 1639
13.549.1 Detailed Description . . . 1640
13.550 gnss_block_interface.h . . . 1640
13.551 observables_interface.h File Reference . . . 1641
13.551.1 Detailed Description . . . 1641
13.552 observables_interface.h . . . 1642
13.553 pvt_interface.h File Reference . . . 1642
13.553.1 Detailed Description . . . 1643
13.554 pvt_interface.h . . . 1643
13.555 signal_source_interface.h . . . 1644
13.556 telemetry_decoder_interface.h File Reference . . . 1645
13.556.1 Detailed Description . . . 1645
13.557 telemetry_decoder_interface.h . . . 1645
13.558 tracking_interface.h File Reference . . . 1646
13.558.1 Detailed Description . . . 1646
13.559 tracking_interface.h . . . 1647
13.560 channel_event.h File Reference . . . 1647
13.560.1 Detailed Description . . . 1648
13.561 channel_event.h . . . 1648

13.562 channel_status_msg_receiver.h File Reference . . . 1649
13.562.1 Detailed Description . . . 1649
13.563 channel_status_msg_receiver.h . . . 1649
13.564 command_event.h File Reference . . . 1650
13.564.1 Detailed Description . . . 1651
13.565 command_event.h . . . 1651
13.566 galileo_e6_has_msg_receiver.h File Reference . . . 1651
13.566.1 Detailed Description . . . 1652
13.567 galileo_e6_has_msg_receiver.h . . . 1652
13.568 galileo_tow_map.h File Reference . . . 1654
13.568.1 Detailed Description . . . 1655
13.569 galileo_tow_map.h . . . 1655
13.570 gnss_crypto.h File Reference . . . 1656
13.570.1 Detailed Description . . . 1656
13.571 gnss_crypto.h . . . 1656
13.572 gnss_sdr_fpga_sample_counter.h File Reference . . . 1658
13.572.1 Detailed Description . . . 1658
13.573 gnss_sdr_fpga_sample_counter.h . . . 1658
13.574 gnss_sdr_log_sink.h File Reference . . . 1659
13.574.1 Detailed Description . . . 1660
13.575 gnss_sdr_log_sink.h . . . 1660
13.576 gnss_sdr_sample_counter.h File Reference . . . 1661
13.576.1 Detailed Description . . . 1661
13.577 gnss_sdr_sample_counter.h . . . 1662
13.578 gnss_sdr_supl_client.h File Reference . . . 1663
13.578.1 Detailed Description . . . 1663
13.579 gnss_sdr_supl_client.h . . . 1663
13.580 gnss_sdr_time_counter.h File Reference . . . 1667
13.580.1 Detailed Description . . . 1667
13.581 gnss_sdr_time_counter.h . . . 1667
13.582 ini.h File Reference . . . 1668
13.582.1 Detailed Description . . . 1668
13.582.1.1 https://github.com/benhoyt/inih . . . 1669
13.583 ini.h . . . 1669
13.584 INIReader.h File Reference . . . 1670
13.584.1 Detailed Description . . . 1670
13.584.1.1 https://github.com/benhoyt/inih . . . 1670
13.585 INIReader.h . . . 1670
13.586 nav_message_monitor.h File Reference . . . 1671
13.586.1 Detailed Description . . . 1672
13.587 nav_message_monitor.h . . . 1672
13.588 nav_message_packet.h File Reference . . . 1673

13.588.1 Detailed Description . . . 1673
13.589 nav_message_packet.h . . . 1673
13.590 nav_message_udp_sink.h File Reference . . . 1674
13.590.1 Detailed Description . . . 1675
13.591 nav_message_udp_sink.h . . . 1675
13.592 osnma_crypto_material.h File Reference . . . 1676
13.592.1 Detailed Description . . . 1676
13.593 osnma_crypto_material.h . . . 1676
13.594 osnma_helper.h File Reference . . . 1677
13.594.1 Detailed Description . . . 1678
13.595 osnma_helper.h . . . 1678
13.596 osnma_msg_receiver.h File Reference . . . 1679
13.596.1 Detailed Description . . . 1679
13.596.2 Macro Definition Documentation . . . 1680
13.596.2.1 FRIEND_TEST . . . 1680
13.597 osnma_msg_receiver.h . . . 1680
13.598 osnma_nav_data_manager.h File Reference . . . 1685
13.598.1 Detailed Description . . . 1685
13.599 osnma_nav_data_manager.h . . . 1685
13.600 protobuf_cleanup_manager.h File Reference . . . 1686
13.600.1 Detailed Description . . . 1686
13.601 protobuf_cleanup_manager.h . . . 1686
13.602 serdes_nav_message.h File Reference . . . 1687
13.602.1 Detailed Description . . . 1687
13.603 serdes_nav_message.h . . . 1687
13.604 string_converter.h File Reference . . . 1689
13.604.1 Detailed Description . . . 1689
13.605 string_converter.h . . . 1689
13.606 uio_fpga.h File Reference . . . 1690
13.606.1 Detailed Description . . . 1690
13.607 uio_fpga.h . . . 1691
13.608 gnss_synchro_monitor.h File Reference . . . 1691
13.608.1 Detailed Description . . . 1692
13.609 gnss_synchro_monitor.h . . . 1692
13.610 gnss_synchro_udp_sink.h File Reference . . . 1693
13.610.1 Detailed Description . . . 1693
13.611 gnss_synchro_udp_sink.h . . . 1694
13.612 serdes_gnss_synchro.h File Reference . . . 1694
13.612.1 Detailed Description . . . 1695
13.613 serdes_gnss_synchro.h . . . 1695
13.614 concurrent_map.h File Reference . . . 1697
13.614.1 Detailed Description . . . 1697

13.615 concurrent_map.h . . . 1698
13.616 concurrent_queue.h File Reference . . . 1699
13.616.1 Detailed Description . . . 1699
13.617 concurrent_queue.h . . . 1699
13.618 control_thread.h File Reference . . . 1701
13.618.1 Detailed Description . . . 1701
13.619 control_thread.h . . . 1701
13.620 file_configuration.h File Reference . . . 1704
13.620.1 Detailed Description . . . 1705
13.621 file_configuration.h . . . 1705
13.622 gnss_block_factory.h File Reference . . . 1706
13.622.1 Detailed Description . . . 1707
13.623 gnss_block_factory.h . . . 1707
13.624 gnss_flowgraph.h File Reference . . . 1708
13.624.1 Detailed Description . . . 1709
13.625 gnss_flowgraph.h . . . 1709
13.626 in_memory_configuration.h File Reference . . . 1713
13.626.1 Detailed Description . . . 1714
13.627 in_memory_configuration.h . . . 1714
13.628 satellite_visibility.h File Reference . . . 1715
13.628.1 Detailed Description . . . 1715
13.629 satellite_visibility.h . . . 1716
13.630 tcp_cmd_interface.h File Reference . . . 1719
13.630.1 Detailed Description . . . 1719
13.631 tcp_cmd_interface.h . . . 1719
13.632 agnss_ref_location.h File Reference . . . 1720
13.632.1 Detailed Description . . . 1721
13.633 agnss_ref_location.h . . . 1721
13.634 agnss_ref_time.h File Reference . . . 1722
13.634.1 Detailed Description . . . 1723
13.635 agnss_ref_time.h . . . 1723
13.636 Beidou_B1C.h File Reference . . . 1724
13.636.1 Detailed Description . . . 1725
13.637 Beidou_B1C.h . . . 1725
13.638 Beidou_B1C_codes.h File Reference . . . 1726
13.638.1 Detailed Description . . . 1726
13.638.2 Variable Documentation . . . 1726
13.638.2.1 BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH . . . 1726
13.638.2.2 BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH . . . 1726
13.639 Beidou_B1C_codes.h . . . 1726
13.640 Beidou_B1C_prn.h File Reference . . . 1727
13.640.1 Detailed Description . . . 1727

13.640.2 Variable Documentation . . . 1727
13.640.2.1 BEIDOU_B1C_DATA_PRN_P . . . 1727
13.640.2.2 BEIDOU_B1C_DATA_PRN_W . . . 1727
13.640.2.3 BEIDOU_B1C_PILOT_PRN_P . . . 1727
13.640.2.4 BEIDOU_B1C_PILOT_PRN_W . . . 1727
13.640.2.5 BEIDOU_B1C_PILOT_SECONDARY_P . . . 1727
13.640.2.6 BEIDOU_B1C_PILOT_SECONDARY_W . . . 1728
13.641 Beidou_B1C_prn.h . . . 1728
13.642 Beidou_B1I.h File Reference . . . 1732
13.642.1 Detailed Description . . . 1733
13.643 Beidou_B1I.h . . . 1733
13.644 Beidou_B2a.h File Reference . . . 1734
13.644.1 Detailed Description . . . 1735
13.645 Beidou_B2a.h . . . 1735
13.646 Beidou_B3I.h File Reference . . . 1735
13.646.1 Detailed Description . . . 1736
13.647 Beidou_B3I.h . . . 1736
13.648 beidou_bdgim.h File Reference . . . 1737
13.648.1 Detailed Description . . . 1737
13.649 beidou_bdgim.h . . . 1737
13.650 Beidou_CNAV1.h File Reference . . . 1738
13.650.1 Detailed Description . . . 1739
13.651 Beidou_CNAV1.h . . . 1739
13.652 beidou_cnav1_ephemeris.h File Reference . . . 1740
13.652.1 Detailed Description . . . 1740
13.653 beidou_cnav1_ephemeris.h . . . 1741
13.654 beidou_cnav1_iono.h File Reference . . . 1742
13.654.1 Detailed Description . . . 1742
13.655 beidou_cnav1_iono.h . . . 1742
13.656 beidou_cnav1_ldpc.h File Reference . . . 1743
13.656.1 Detailed Description . . . 1743
13.657 beidou_cnav1_ldpc.h . . . 1744
13.658 beidou_cnav1_navigation_message.h File Reference . . . 1745
13.658.1 Detailed Description . . . 1746
13.659 beidou_cnav1_navigation_message.h . . . 1746
13.660 beidou_cnav1_utc_model.h File Reference . . . 1748
13.660.1 Detailed Description . . . 1748
13.661 beidou_cnav1_utc_model.h . . . 1748
13.662 Beidou_CNAV2.h File Reference . . . 1749
13.662.1 Detailed Description . . . 1749
13.663 Beidou_CNAV2.h . . . 1750
13.664 beidou_cnav2_ldpc.h File Reference . . . 1750

13.664.1 Detailed Description . . . 1751
13.665 beidou_cnav2_ldpc.h . . . 1751
13.666 beidou_cnav2_navigation_message.h File Reference . . . 1752
13.666.1 Detailed Description . . . 1752
13.667 beidou_cnav2_navigation_message.h . . . 1752
13.668 Beidou_DNAV.h File Reference . . . 1753
13.668.1 Detailed Description . . . 1758
13.669 Beidou_DNAV.h . . . 1758
13.670 beidou_dnav_almanac.h File Reference . . . 1762
13.670.1 Detailed Description . . . 1762
13.671 beidou_dnav_almanac.h . . . 1762
13.672 beidou_dnav_differential_corrections.h File Reference . . . 1763
13.672.1 Detailed Description . . . 1764
13.673 beidou_dnav_differential_corrections.h . . . 1764
13.674 beidou_dnav_ephemeris.h File Reference . . . 1765
13.674.1 Detailed Description . . . 1765
13.675 beidou_dnav_ephemeris.h . . . 1765
13.676 beidou_dnav_iono.h File Reference . . . 1767
13.676.1 Detailed Description . . . 1767
13.677 beidou_dnav_iono.h . . . 1767
13.678 beidou_dnav_navigation_message.h File Reference . . . 1768
13.678.1 Detailed Description . . . 1769
13.679 beidou_dnav_navigation_message.h . . . 1769
13.680 beidou_dnav_utc_model.h File Reference . . . 1774
13.680.1 Detailed Description . . . 1774
13.681 beidou_dnav_utc_model.h . . . 1774
13.682 beidou_ldpc.h File Reference . . . 1775
13.682.1 Detailed Description . . . 1776
13.682.2 Function Documentation . . . 1776
13.682.2.1 add() . . . 1776
13.682.2.2 inv() . . . 1776
13.682.2.3 mul() . . . 1776
13.682.2.4 valid_symbol() . . . 1776
13.682.3 Variable Documentation . . . 1777
13.682.3.1 BEIDOU_GF64_EXP . . . 1777
13.682.3.2 BEIDOU_GF64_LOG . . . 1777
13.682.3.3 BEIDOU_LDPC_MAX_ITER . . . 1777
13.682.3.4 BEIDOU_LDPC_NM . . . 1777
13.682.3.5 kFieldSize . . . 1777
13.682.3.6 kOrder . . . 1777
13.682.3.7 kZero . . . 1777
13.683 beidou_ldpc.h . . . 1777

13.684 display.h File Reference . . . 1779
13.684.1 Detailed Description . . . 1779
13.685 display.h . . . 1780
13.686 galileo_almanac.h File Reference . . . 1781
13.686.1 Detailed Description . . . 1781
13.687 galileo_almanac.h . . . 1781
13.688 galileo_almanac_helper.h File Reference . . . 1782
13.688.1 Detailed Description . . . 1782
13.688.1.1 autotoc_md684 . . . 1782
13.689 galileo_almanac_helper.h . . . 1782
13.690 Galileo_CNAV.h File Reference . . . 1784
13.690.1 Detailed Description . . . 1786
13.691 Galileo_CNAV.h . . . 1786
13.692 galileo_cnav_message.h File Reference . . . 1788
13.692.1 Detailed Description . . . 1788
13.693 galileo_cnav_message.h . . . 1789
13.694 Galileo_E1.h File Reference . . . 1790
13.694.1 Detailed Description . . . 1791
13.695 Galileo_E1.h . . . 1791
13.696 Galileo_E5a.h File Reference . . . 1808
13.696.1 Detailed Description . . . 1809
13.697 Galileo_E5a.h . . . 1809
13.698 Galileo_E5b.h File Reference . . . 1852
13.698.1 Detailed Description . . . 1853
13.699 Galileo_E5b.h . . . 1853
13.700 Galileo_E6.h File Reference . . . 1895
13.700.1 Detailed Description . . . 1896
13.701 Galileo_E6.h . . . 1896
13.702 galileo_ephemeris.h File Reference . . . 1918
13.702.1 Detailed Description . . . 1918
13.703 galileo_ephemeris.h . . . 1919
13.704 Galileo_FNAV.h File Reference . . . 1921
13.704.1 Detailed Description . . . 1924
13.705 Galileo_FNAV.h . . . 1924
13.706 galileo_fnav_message.h File Reference . . . 1926
13.706.1 Detailed Description . . . 1926
13.707 galileo_fnav_message.h . . . 1927
13.708 galileo_has_data.h File Reference . . . 1930
13.708.1 Detailed Description . . . 1930
13.709 galileo_has_data.h . . . 1931
13.710 galileo_has_page.h File Reference . . . 1933
13.710.1 Detailed Description . . . 1934

13.711 galileo_has_page.h . . . 1934
13.712 Galileo_INAV.h File Reference . . . 1934
13.712.1 Detailed Description . . . 1939
13.713 Galileo_INAV.h . . . 1939
13.714 galileo_inav_message.h File Reference . . . 1943
13.714.1 Detailed Description . . . 1943
13.715 galileo_inav_message.h . . . 1944
13.716 galileo_iono.h File Reference . . . 1950
13.716.1 Detailed Description . . . 1950
13.717 galileo_iono.h . . . 1951
13.718 galileo_ism.h File Reference . . . 1952
13.718.1 Detailed Description . . . 1952
13.719 galileo_ism.h . . . 1952
13.720 Galileo_OSNMA.h File Reference . . . 1955
13.720.1 Detailed Description . . . 1956
13.721 Galileo_OSNMA.h . . . 1956
13.722 galileo_reduced_ced.h File Reference . . . 1958
13.722.1 Detailed Description . . . 1958
13.723 galileo_reduced_ced.h . . . 1959
13.724 galileo_utc_model.h File Reference . . . 1960
13.724.1 Detailed Description . . . 1960
13.725 galileo_utc_model.h . . . 1960
13.726 glonass_gnav_almanac.h File Reference . . . 1961
13.726.1 Detailed Description . . . 1962
13.727 glonass_gnav_almanac.h . . . 1962
13.728 glonass_gnav_ephemeris.h File Reference . . . 1964
13.728.1 Detailed Description . . . 1964
13.729 glonass_gnav_ephemeris.h . . . 1964
13.730 glonass_gnav_navigation_message.h File Reference . . . 1967
13.730.1 Detailed Description . . . 1968
13.731 glonass_gnav_navigation_message.h . . . 1968
13.732 glonass_gnav_utc_model.h File Reference . . . 1971
13.732.1 Detailed Description . . . 1971
13.733 glonass_gnav_utc_model.h . . . 1971
13.734 GLONASS_L1_L2_CA.h File Reference . . . 1973
13.734.1 Detailed Description . . . 1977
13.735 GLONASS_L1_L2_CA.h . . . 1977
13.736 gnss_almanac.h File Reference . . . 1982
13.736.1 Detailed Description . . . 1982
13.737 gnss_almanac.h . . . 1982
13.738 gnss_ephemeris.h File Reference . . . 1983
13.738.1 Detailed Description . . . 1984

13.739 gnss_ephemeris.h . . . 1984
13.740 gnss_frequencies.h File Reference . . . 1985
13.740.1 Detailed Description . . . 1986
13.741 gnss_frequencies.h . . . 1986
13.742 gnss_obs_codes.h File Reference . . . 1987
13.742.1 Detailed Description . . . 1990
13.743 gnss_obs_codes.h . . . 1990
13.744 gnss_satellite.h File Reference . . . 1991
13.744.1 Detailed Description . . . 1991
13.745 gnss_satellite.h . . . 1991
13.746 gnss_signal.h File Reference . . . 1993
13.746.1 Detailed Description . . . 1993
13.747 gnss_signal.h . . . 1993
13.748 gnss_synchro.h File Reference . . . 1994
13.748.1 Detailed Description . . . 1994
13.749 gnss_synchro.h . . . 1994
13.750 gps_acq_assist.h File Reference . . . 1997
13.750.1 Detailed Description . . . 1998
13.751 gps_acq_assist.h . . . 1998
13.752 gps_almanac.h File Reference . . . 1999
13.752.1 Detailed Description . . . 1999
13.753 gps_almanac.h . . . 1999
13.754 GPS_CNAV.h File Reference . . . 2000
13.754.1 Detailed Description . . . 2003
13.755 GPS_CNAV.h . . . 2003
13.756 gps_cnav_eop.h File Reference . . . 2005
13.756.1 Detailed Description . . . 2005
13.757 gps_cnav_eop.h . . . 2006
13.758 gps_cnav_ephemeris.h File Reference . . . 2006
13.758.1 Detailed Description . . . 2006
13.759 gps_cnav_ephemeris.h . . . 2007
13.760 gps_cnav_iono.h File Reference . . . 2009
13.760.1 Detailed Description . . . 2009
13.761 gps_cnav_iono.h . . . 2009
13.762 gps_cnav_navigation_message.h File Reference . . . 2010
13.762.1 Detailed Description . . . 2010
13.763 gps_cnav_navigation_message.h . . . 2010
13.764 gps_cnav_utc_model.h File Reference . . . 2012
13.764.1 Detailed Description . . . 2012
13.765 gps_cnav_utc_model.h . . . 2012
13.766 gps_ephemeris.h File Reference . . . 2013
13.766.1 Detailed Description . . . 2013

13.767 gps_ephemeris.h . . . 2013
13.768 gps_iono.h File Reference . . . 2015
13.768.1 Detailed Description . . . 2016
13.769 gps_iono.h . . . 2016
13.770 GPS_L1_CA.h File Reference . . . 2017
13.770.1 Detailed Description . . . 2021
13.771 GPS_L1_CA.h . . . 2022
13.772 GPS_L2C.h File Reference . . . 2025
13.772.1 Detailed Description . . . 2026
13.773 GPS_L2C.h . . . 2026
13.774 GPS_L5.h File Reference . . . 2027
13.774.1 Detailed Description . . . 2028
13.775 GPS_L5.h . . . 2029
13.776 gps_navigation_message.h File Reference . . . 2031
13.776.1 Detailed Description . . . 2031
13.777 gps_navigation_message.h . . . 2031
13.778 gps_utc_model.h File Reference . . . 2035
13.778.1 Detailed Description . . . 2035
13.779 gps_utc_model.h . . . 2036
13.780 gps_week_rollover.h File Reference . . . 2037
13.780.1 Detailed Description . . . 2037
13.781 gps_week_rollover.h . . . 2037
13.782 MATH_CONSTANTS.h File Reference . . . 2038
13.782.1 Detailed Description . . . 2041
13.783 MATH_CONSTANTS.h . . . 2041
13.784 osnma_data.h File Reference . . . 2043
13.784.1 Detailed Description . . . 2044
13.784.2 Variable Documentation . . . 2044
13.784.2.1 GALILEO_SECONDS_PER_WEEK . . . 2044
13.784.2.2 OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS . . . 2044
13.784.2.3 OSNMA_AUTH_IOD_MAX_AGE_SECONDS . . . 2044
13.785 osnma_data.h . . . 2044
13.786 osnma_dsm_reader.h File Reference . . . 2048
13.786.1 Detailed Description . . . 2048
13.787 osnma_dsm_reader.h . . . 2048
13.788 qzss.h File Reference . . . 2050
13.788.1 Detailed Description . . . 2050
13.789 qzss.h . . . 2050
13.790 qzss_cnav_eop.h File Reference . . . 2051
13.790.1 Detailed Description . . . 2051
13.791 qzss_cnav_eop.h . . . 2052
13.792 qzss_cnav_iono.h File Reference . . . 2052

13.792.1 Detailed Description . . . 2052
13.793 qzss_cnav_iono.h . . . 2052
13.794 qzss_cnav_utc_model.h File Reference . . . 2053
13.794.1 Detailed Description . . . 2053
13.795 qzss_cnav_utc_model.h . . . 2053
13.796 qzss_iono.h File Reference . . . 2054
13.796.1 Detailed Description . . . 2054
13.797 qzss_iono.h . . . 2054
13.798 qzss_utc_model.h File Reference . . . 2055
13.798.1 Detailed Description . . . 2055
13.799 qzss_utc_model.h . . . 2055
13.800 reed_solomon.h File Reference . . . 2056
13.800.1 Detailed Description . . . 2056
13.801 reed_solomon.h . . . 2056
13.802 sbas_ephemeris.h File Reference . . . 2058
13.802.1 Detailed Description . . . 2059
13.803 sbas_ephemeris.h . . . 2059
13.804 SBAS_L1.h File Reference . . . 2060
13.804.1 Detailed Description . . . 2060
13.805 SBAS_L1.h . . . 2061
13.806 sbas_raw_message.h File Reference . . . 2062
13.806.1 Detailed Description . . . 2062
13.807 sbas_raw_message.h . . . 2062
13.808 signal_flag.h File Reference . . . 2063
13.808.1 Detailed Description . . . 2063
13.808.2 Enumeration Type Documentation . . . 2063
13.808.2.1 signal_flag . . . 2063
13.809 signal_flag.h . . . 2064
13.810 tow_to_trk.h File Reference . . . 2064
13.810.1 Detailed Description . . . 2064
13.811 tow_to_trk.h . . . 2064

Index **2067**

Chapter 1

GNSS-SDR



Figure 1.1 GNSS-SDR logo

Welcome to GNSS-SDR!

GNSS-SDR is an open-source GNSS software receiver freely available to the research community. This project provides a common framework for GNSS signal processing which can operate in a variety of computer platforms. This tool is intended to foster collaboration, increase awareness, and reduce development costs in the field of GNSS receiver design and customized use of GNSS signals.

For details about GNSS-SDR and using it, please see the **main project page** or browse **the source code at GitHub**. You could be also interested in **subscribing to the mailing list**.

1.1 Contents

- Overview
- Building GNSS-SDR
- Using GNSS-SDR
- Control plane
- Signal Processing plane
- About the software license
- Publications and Credits
- Ok, now what?

More details on GNSS-SDR signal processing blocks:

- Signal Source
- Signal Conditioner

- Channel
  - Acquisition
  - Tracking
  - Decoding of the navigation message
- Observables
- Computation of Position, Velocity and Time

1.2 Overview

GNSS-SDR provides an interface to different suitable RF front-ends and implements all the receiver chain up to the navigation solution. Its design allows any kind of customization, including interchangeability of signal sources, signal processing algorithms, interoperability with other systems, output formats, and offers interfaces to all the intermediate signals, parameters and variables. The goal is to write efficient and truly reusable code, easy to read and maintain, with fewer bugs, and producing highly optimized executables in a variety of hardware platforms and operating systems. In that sense, the challenge consists of defining a gentle balance within level of abstraction and performance. GNSS-SDR runs in a personal computer and provides interfaces through USB and Ethernet buses to a variety of either commercially available or custom-made RF front-ends, adapting the processing algorithms to different sampling frequencies, intermediate frequencies and sample resolutions. This makes possible rapid prototyping of specific receivers intended, for instance, to geodetic applications, observation of the ionospheric impact on navigation signals, GNSS reflectometry, signal quality monitoring, or carrier-phase based navigation techniques.

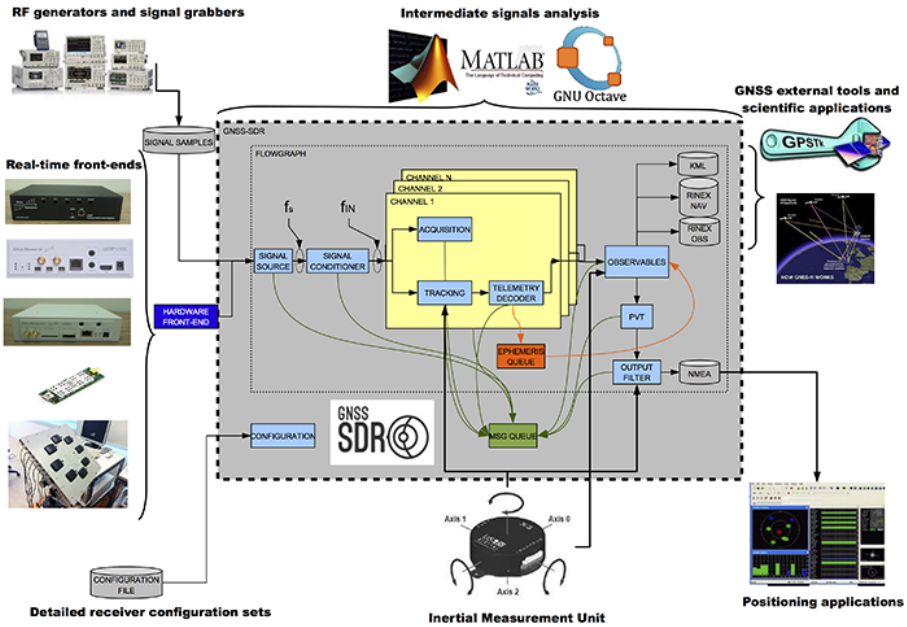


Figure 1.2 Overview

As signal inputs, it accepts:

- Raw data file captured with a data grabber (digitized at some intermediate frequency or directly at baseband).
- Any suitable RF configuration that can be driven by the Universal Software Radio Peripheral Hardware Driver (UHD). This includes all current and future Ettus Research products. The USRP1 + DBSRX 2.2 daughterboard is an example of working configuration for GPS L1 C/A and Galileo E1B and E1C signals.

- Experimentally, with some USB DVB-T dongles based on the Realtek RTL2832U chipset.
- For mathematical representations of the targeted signals, check out the Signal model page.

As outputs, it provides:

- Dump of intermediate signals (configurable by the user)
- The processing is logged at a system temporary folder (usually, /tmp)
- Observables in form of RINEX file (experimental)
- Navigation message data in form of RINEX file
- Position, Velocity and Time solution in KML format and NMEA

1.3 Building GNSS-SDR

In principle, GNSS-SDR can be built in any Unix-like system. In practice, it depends on being able to install all the required dependencies. See the building guide page for details about the project's dependencies and build process. Mainly, it consists on installing GNU Radio plus some few more libraries:

- Armadillo, a C++ linear algebra library,
- Boost, a set of free peer-reviewed portable C++ source libraries,
- Gflags, a library that implements commandline flags processing,
- Glog, a library that implements application-level logging,
- Googletest, Google's framework for writing C++ tests,
- Mako, a template library written in Python,
- Matio, a MATLAB MAT File I/O Library,
- Protocol Buffers, a language-neutral, platform-neutral extensible mechanism for serializing structured data,
- PugiXML, a light-weight, simple and fast XML parser for C++ with XPath support,
- Volk, a Vector-Optimized Library of Kernels which provides an abstraction of optimized math routines targeting several SIMD processors,

and, optionally,

- GNU Radio modules for hardware interface (gr-uhd, gr-osmosdr, gr-iio),
- Benchmark, a library to benchmark code snippets,
- Gperftools, which provides fast, multi-threaded malloc() and performance analysis tools.

After all dependencies are installed, clone the GNSS-SDR repository:

```
$ git clone https://github.com/gnss-sdr/gnss-sdr
```

This will create a folder named gnss-sdr with the following structure:

```
|-gnss-sdr
|---cmake       <- CMake-related files
|---conf        <- Configuration files. Each file represents one receiver.
|---docs        <- Contains documentation-related files
|---install     <- Executables
|---src         <- Source code folder
|-----algorithms
|-------PVT
|-------acquisition
|-------channel
|-------conditioner
|-------data_type_adapter
|-------input_filter
|-------libs
|-------observables
|-------resampler
|-------signal_source
|-------telemetry_decoder
|-------tracking
|-----core
|-------interfaces
|-------libs
|-------receiver
|-------system_parameters
|-----main
|---tests
|---utils     <- some utilities (e.g. Matlab scripts)
```

You are now ready to build GNSS-SDR by using CMake as building tool:

```
$ cd gnss-sdr && mkdir build && cd build
$ cmake ..
$ make
```

If everything goes well, three new executables will be created at `gnss-sdr/install`, namely `gnss-sdr`, `volk_gnsssdr_profile` and `run_tests`. You can run them from that folder, but if you prefer to install gnss-sdr on your system and have it available anywhere else, do:

```
$ sudo make install
```

This will make a copy of the conf/ folder into /usr/local/share/gnss-sdr/conf for your reference. We suggest to create a working directory at your preferred location and store your own configuration and data files there.

You can create the documentation by doing:

```
$ make doc
```

from the building folder. In both cases, Doxygen will generate HTML documentation that can be retrieved pointing your browser of preference to `gnss-sdr/docs/html/index.html`.

There are two more extra targets available. In the building folder:

```
$ make doc-clean
```

will remove the content of previously-generated documentation and, if a LaTeX installation is detected in your system,

```
$ make pdfmanual
```

will create a PDF manual at `gnss-sdr/docs/GNSS-SDR_manual.pdf`. Please note that the PDF generation requires some fonts to be installed on the host system. In Ubuntu, those fonts do not come by default. You can install them by doing:

```
$ sudo apt install texlive-fonts-recommended
```

and then run `cmake ../` and `make pdfmanual` again.

1.3.1 Debug and Release builds

By default, CMake will build the Release version, meaning that the compiler will generate a faster, optimized executable. This is the recommended build type when using a RF front-end and you need to attain real time. If working with a file (and thus without real-time constraints), you may want to obtain more information about the internals of the receiver, as well as more fine-grained logging. This can be done by building the Debug version, by doing:

```
$ cd gnss-sdr && mkdir build-debug && cd build-debug
$ cmake -DCMAKE_BUILD_TYPE=Debug ..
$ make
```

1.3.2 Updating GNSS-SDR

If you checked out GNSS-SDR some days ago, it is possible that some developer had updated files at the Git repository. You can update your local copy by doing:

```
$ git checkout next
$ git pull https://github.com/gnss-sdr/gnss-sdr next
```

Before rebuilding the source code, it is safe (and recommended) to remove the remainders of old builds:

```
$ cd <building folder>
$ sudo make uninstall  ; if you installed it before
$ rm -rf *
```

You can also check The Git Book for more information about Git usage.

1.4 Using GNSS-SDR

With GNSS-SDR, you can define you own receiver, work with captured raw data or from a RF front-end, dump into files intermediate signals, or tune every single algorithm used in the Signal Processing plane. All the configuration is done in a single file. Those configuration files reside at the `gnss-sdr/conf` folder. By default, the executable `gnss-sdr` will read the configuration available at `gnss-sdr/conf/gnss-sdr.conf`. You can edit that file to fit your needs, or even better, define a new `my_receiver.conf` file with your own configuration. This new receiver can be done by invoking gnss-sdr with the `-config_file` flag pointing to your configuration file:

```
$ gnss-sdr --config_file=../conf/my_receiver.conf
```

You can see a guide of available implementations at the online documentation. That folder contains other working examples as well. If you have a working configuration and want to share it will others, please email it to the **GNSS-SDR developers mailing list** and we will be happy to upload it to the server.

You can use a single configuration file for processing different data files, specifying the file to be processed with the `-signal_source` flag:

```
$ gnss-sdr --config_file=../conf/my_receiver.conf --signal_source=./my_captured_data.dat
```

This will override the `SignalSource.filename` specified in the configuration file.

You can get a complete list of available commandline flags by doing:

```
$ gnss-sdr --help
```

1.5 Control plane

GNSS-SDR's main method initializes the logging library, processes the command line flags, if any, provided by the user and instantiates a ControlThread object. Its constructor reads the configuration file, creates a control queue and creates a flowgraph according to the configuration. Then, the program's main method calls the run() method of the instantiated object, an action that connects the flowgraph and starts running it. After that, and until a stop message is received, it reads control messages sent by the receiver's modules through a safe-thread queue and processes them. Finally, when a stop message is received, the main method executes the destructor of the ControlThread object, which deallocates memory, does other cleanup and exits the program.

The GNSSFlowgraph class is responsible for preparing the graph of blocks according to the configuration, running it, modifying it during run-time and stopping it. Blocks are identified by its role. This class knows which roles it has to instantiate and how to connect them. It relies on the configuration to get the correct instances of the roles it needs and then it applies the connections between GNU Radio blocks to make the graph ready to be started. The complexity related to managing the blocks and the data stream is handled by GNU Radio's `gr::top_block` class. GNSSFlowgraph wraps the `gr::top_block` instance so we can take advantage of the GNSS block factory, the configuration system and the processing blocks. This class is also responsible for applying changes to the configuration of the flowgraph during run-time, dynamically reconfiguring channels: it selects the strategy for selecting satellites. This can range from a sequential search over all the satellites' ID to smarter approaches that determine what are the satellites most likely in-view based on rough estimations of the receiver position in order to avoid searching satellites in the other side of the Earth.

The Control Plane is in charge of creating a flowgraph according to the configuration and then managing the modules. Configuration allows users to define in an easy way their own custom receiver by specifying the flowgraph (type of signal source, number of channels, algorithms to be used for each channel and each module, strategies for satellite selection, type of output format, etc.). Since it is difficult to foresee what future module implementations will be needed in terms of configuration, we used a very simple approach that can be extended without a major impact in the code. This can be achieved by simply mapping the names of the variables in the modules with the names of the parameters in the configuration.

1.5.1 Configuration

Properties are passed around within the program using the ConfigurationInterface class. There are two implementations of this interface: FileConfiguration and InMemoryConfiguration. FileConfiguration reads the properties (pairs of property name and value) from a file and stores them internally. InMemoryConfiguration does not read from a file; it remains empty after instantiation and property values and names are set using the set property method. FileConfiguration is intended to be used in the actual GNSS-SDR application whereas InMemoryConfiguration is intended to be used in tests to avoid file-dependency in the file system. Classes that need to read configuration parameters will receive instances of ConfigurationInterface from where they will fetch the values. For instance, parameters related to SignalSource should look like this:

```
SignalSource.parameter1=value1
SignalSource.parameter2=value2
```

The name of these parameters can be anything but one reserved word: implementation. This parameter indicates in its value the name of the class that has to be instantiated by the factory for that role. For instance, if our signal source is providing data already at baseband and thus we want to use the implementation Pass_Through for module SignalConditioner, the corresponding line in the configuration file would be

```
SignalConditioner.implementation=Pass_Through
```

Since the configuration is just a set of property names and values without any meaning or syntax, the system is very versatile and easily extendable. Adding new properties to the system only implies modifications in the classes that will make use of these properties. In addition, the configuration files are not checked against any strict syntax so it is always in a correct status (as long as it contains pairs of property names and values in `INI format`).

1.5.2 GNSS block factory

Hence, the application defines a simple accessor class to fetch the configuration pairs of values and passes them to a factory class called GNSSBlockFactory. This factory decides, according to the configuration, which class needs to be instantiated and which parameters should be passed to the constructor. Hence, the factory encapsulates the complexity of blocks' instantiation. With that approach, adding a new block that requires new parameters will be as simple as adding the block class and modifying the factory to be able to instantiate it. This loose coupling between the blocks' implementations and the syntax of the configuration enables extending the application capacities in a high degree. It also allows to produce fully customized receivers, for instance a testbed for acquisition algorithms, and to place observers at any point of the receiver chain.

1.6 Signal Processing plane

GNU Radio's class `gr::basic_block` is the abstract base class for all signal processing blocks, a bare abstraction of an entity that has a name and a set of inputs and outputs. It is never instantiated directly; rather, this is the abstract parent class of both `gr::hier_block2`, which is a recursive container that adds or removes processing or hierarchical blocks to the internal graph, and `gr::block`, which is the abstract base class for all the processing blocks.

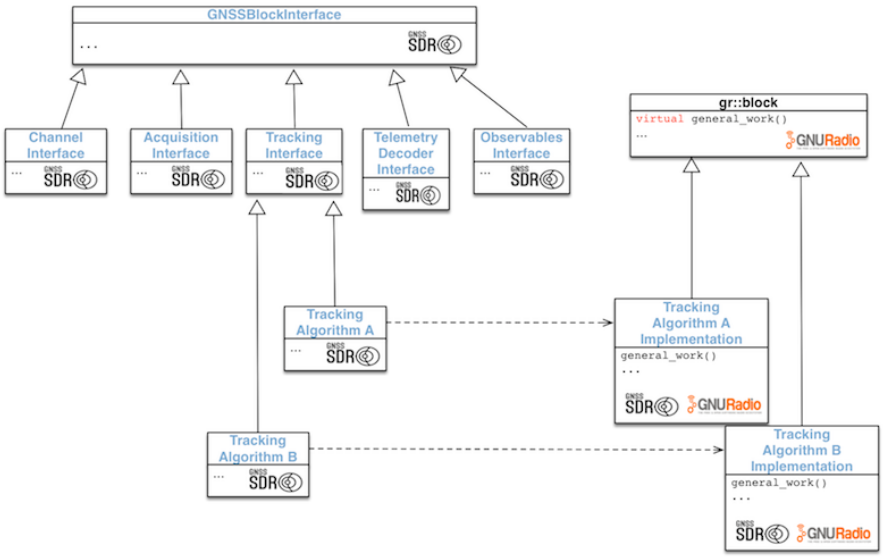


Figure 1.3 Class hierarchy of signal processing blocks

A signal processing flow is constructed by creating a tree of hierarchical blocks, which at any level may also contain terminal nodes that actually implement signal processing functions.

Class `gr::top_block` is the top-level hierarchical block representing a flowgraph. It defines GNU Radio runtime functions used during the execution of the program: run(), start(), stop(), wait(), etc. A a subclass called GNSSBlockInterface is the common interface for all the GNSS-SDR modules. It defines pure virtual methods, that are required to be implemented by a derived class.

Subclassing GNSSBlockInterface, we defined interfaces for the GNSS receiver blocks depicted in the figure above. This hierarchy provides the definition of different algorithms and different implementations, which will be instantiated according to the configuration. This strategy allows multiple implementations sharing a common interface, achieving the objective of decoupling interfaces from implementations: it defines a family of algorithms, encapsulates each one, and makes them interchangeable. Hence, we let the algorithm vary independently from the program that uses it.

1.6.1 Signal Source

The input of a software receiver are the raw bits that come out from the front-end's analog-to-digital converter (ADC). Those bits can be read from a file stored in the hard disk or directly in real-time from a hardware device through USB or Ethernet buses.

The Signal Source module is in charge of implementing the hardware driver, that is, the portion of the code that communicates with the RF front-end and receives the samples coming from the ADC. This communication is usually performed through USB or Ethernet buses. Since real-time processing requires a highly optimized implementation of the whole receiver, this module also allows to read samples from a file stored in a hard disk, and thus processing without time constraints. Relevant parameters of those samples are the intermediate frequency (or baseband I&Q components), the sampling rate and number of bits per sample, that must be specified by the user in the configuration file.

This module also performs bit-depth adaptation, since most of the existing RF front-ends provide samples quantized with 2 or 3 bits, while operations inside the processor are performed on 32- or 64-bit words, depending on its architecture. Although there are implementations of the most intensive computational processes (mainly correlation) that take advantage of specific data types and architectures for the sake of efficiency, the approach is processor-specific and hardly portable. We suggest to keep signal samples in standard data types and letting the compiler select the best library version (implemented using SIMD or any other processor-specific technology) of the required routines for a given processor.

Example: FileSignalSource

The user can configure the receiver for reading from a file, setting in the configuration file the data file location, sample format, and the sampling frequency and intermediate frequency at what the signal was originally captured.

```
;######### SIGNAL_SOURCE CONFIG ############
SignalSource.implementation=File_Signal_Source
SignalSource.filename=/home/user/gnss-sdr/data/my_capture.dat
SignalSource.item_type=gr_complex
SignalSource.sampling_frequency=4000000 ; Sampling frequency in [Hz]
SignalSource.freq=1575420000 ; RF front-end center frequency in [Hz]
```

Example: UhdSignalSource

The user may prefer to use a UHD-compatible RF front-end and try real-time processing. For instance, for a USRP1 + DBSRX daughterboard, use:

```
;######### SIGNAL_SOURCE CONFIG ############
SignalSource.implementation=UHD_Signal_Source
SignalSource.item_type=gr_complex
SignalSource.sampling_frequency=4000000 ; Sampling frequency in [Hz]
SignalSource.freq=1575420000 ; RF front-end center frequency in [Hz]
SignalSource.gain=60 ; Front-end gain in dB
SignalSource.subdevice=B:0 ; UHD subdevice specification (for USRP1 use A:0 or B:0)
```

Other examples are available at `gnss-sdr/conf`.

1.6.2 Signal Conditioner

The signal conditioner is in charge of resampling the signal and delivering a reference sample rate to the downstream processing blocks, acting as a facade between the signal source and the synchronization channels, providing a simplified interface to the input signal. In case of multiband front-ends, this module would be in charge of providing a separated data stream for each band.

1.6.3 Channel

A channel encapsulates all signal processing devoted to a single satellite. Thus, it is a large composite object which encapsulates the Acquisition, Tracking and Decoding of the navigation message modules. As a composite object, it can be treated as a single entity, meaning that it can be easily replicated. Since the number of channels is selectable by the user in the configuration file, this approach helps improving the scalability and maintainability of the receiver.

This module is also in charge of managing the interplay between acquisition and tracking. Acquisition can be initialized in several ways, depending on the prior information available (called cold start when the receiver has no information about its position nor the satellites almanac; warm start when a rough location and the approximate time of day are available, and the receiver has a recently recorded almanac broadcast; or hot start when the receiver was tracking a satellite and the signal line of sight broke for a short period of time, but the ephemeris and almanac data is still valid, or this information is provided by other means), and an acquisition process can finish deciding that the satellite is not present, that longer integration is needed in order to confirm the presence of the satellite, or declaring the satellite present. In the latter case, acquisition process should stop and trigger the tracking module with coarse estimations of the synchronization parameters.

The abstract class ChannelInterface represents an interface to a channel GNSS block. Check Channel for an actual implementation.

1.6.3.1 Acquisition

The first task of a GNSS receiver is to detect the presence or absence of in-view satellites. This is done by the acquisition system process, which also provides a coarse estimation of two signal parameters: the frequency shift with respect to the nominal IF frequency, and a delay term which allows the receiver to create a local code aligned with the incoming code. AcquisitionInterface is the common interface for all the acquisition algorithms and their corresponding implementations. Algorithms' interface, that may vary depending on the use of information external to the receiver, such as in Assisted GNSS, is defined in classes referred to as *adapters*. These adapters wrap the GNU Radio blocks interface into a compatible interface expected by AcquisitionInterface. This allows the use of existing GNU Radio blocks derived from `gr::block`, and ensures that newly developed implementations will also be reusable in other GNU Radio-based applications. Moreover, it adds still another layer of abstraction, since each given acquisition algorithm can have different implementations (for instance using different numerical libraries). In such a way, implementations can be continuously improved without having any impact neither on the algorithm interface nor the general acquisition interface.

Check PcpsAcquisitionAdapter for an example of an adapter from a Parallel Code Phase Search (PCPS) acquisition block to AcquisitionInterface, and pcps_acquisition for an example of a block implementation. The source code of all the available acquisition algorithms is located at:

```
|-gnss-sdr
|---src
|-----algorithms
|-------acquisition
|---------adapters          <- Adapters of the processing blocks to an AcquisitionInterface
|---------gnuradio_blocks   <- Signal processing blocks implementation
```

The user can select a given implementation for the algorithm to be used in each receiver channel, as well as their parameters, in the configuration file:

```
;######### ACQUISITION GLOBAL CONFIG ############

;#implementation: Acquisition algorithm selection for this channel:
Acquisition_1C.implementation=GPS_L1_CA_PCPS_Acquisition
;#dump: Enable or disable the acquisition internal data file logging [true] or [false]
Acquisition_1C.dump=false
;#filename: Log path and filename
Acquisition_1C.dump_filename=./acq_dump.dat
```

```
;#item_type: Type and resolution for each of the signal samples. Use only gr_complex in this version.
Acquisition_1C.item_type=gr_complex
;#coherent_integration_time_ms: Signal block duration for the acquisition signal detection [ms]
Acquisition_1C.coherent_integration_time_ms=1
;#threshold: Acquisition threshold
Acquisition_1C.threshold=2.5
;#pfa: Acquisition false alarm probability. This option overrides the threshold option.
Acquisition_1C.pfa=0.0001
;#doppler_max: Maximum expected Doppler shift [Hz]
Acquisition_1C.doppler_max=5000
;#doppler_step: Doppler step in the grid search [Hz]
Acquisition_1C.doppler_step=250
```

1.6.3.2 Tracking

When a satellite is declared present, the parameters estimated by the acquisition module are then fed to the receiver tracking module, which represents the second stage of the signal processing unit, aiming to perform a local search for accurate estimates of code delay and carrier phase, and following their eventual variations.

Again, a class hierarchy consisting of a TrackingInterface class and subclasses implementing algorithms provides a way of testing different approaches, with full access to their parameters. Check DllPllTrackingAdapter for an example of an adapter from a DLL/PLL tracking block to TrackingInterface, and dll_pll_veml_tracking for an example of a signal processing block implementation. There are also available some useful classes and functions for signal tracking; take a look at Correlator, lock_detectors.h, tracking_discriminators.h or tracking_2nd_DLL_filter.h.

The source code of all the available tracking algorithms is located at:

```
|-gnss-sdr
|---src
|-----algorithms
|-------tracking
|---------adapters          <- Adapters of the processing blocks to a TrackingInterface
|---------gnuradio_blocks   <- Signal processing blocks implementation
|---------libs              <- libraries of tracking objects (e.g. correlators, discriminators, and so on)
```

The user can select a given implementation for the algorithm to be used in all the tracking blocks, as well as its parameters, in the configuration file:

```
;######### TRACKING GLOBAL CONFIG ############

;#implementation: Selected tracking algorithm
Tracking_1C.implementation=GPS_L1_CA_DLL_PLL_Tracking
;#item_type: Type and resolution for each of the signal samples.
Tracking_1C.item_type=gr_complex

;#dump: Enable or disable the Tracking internal binary data file logging [true] or [false]
Tracking_1C.dump=false

;#dump_filename: Log path and filename. Notice that the tracking channel will add "x.dat" where x is the chann
Tracking_1C.dump_filename=./tracking_ch_

;#pll_bw_hz: PLL loop filter bandwidth [Hz]
Tracking_1C.pll_bw_hz=50.0;

;#dll_bw_hz: DLL loop filter bandwidth [Hz]
Tracking_1C.dll_bw_hz=2.0;

;#fll_bw_hz: FLL loop filter bandwidth [Hz]
Tracking_1C.fll_bw_hz=10.0;

Tracking_1C.pll_filter_order=3 ; PLL loop filter order [2] or [3]
Tracking_1C.dll_filter_order=2 ; DLL loop filter order [1], [2] or [3]

;#early_late_space_chips: correlator early-late space [chips].
Tracking_1C.early_late_space_chips=0.5;
```

1.6.3.3 Decoding of the navigation message

Most of GNSS signal links are modulated by a navigation message containing the time the message was transmitted, orbital parameters of satellites (also known as ephemeris) and an almanac (information about the general system health, rough orbits of all satellites in the network as well as data related to error correction). Navigation data bits are structured in words, pages, subframes, frames and superframes. Sometimes, bits corresponding to a single parameter are spread over different words, and values extracted from different frames are required for proper decoding. Some words are for synchronization purposes, others for error control an others contain actual information. There are also error control mechanisms, from parity checks to forward error correction (FEC) encoding and interleaving, depending on the system.

The common interface is TelemetryDecoderInterface. Check TelemetryDecoderAdapter for an example of an adapter from a telemetry decoder block to TelemetryDecoderInterface, and gps_l1_ca_telemetry_decoder_gs for an actual implementation of a signal processing block. Configuration example:

```
;######### TELEMETRY DECODER CONFIG ############
TelemetryDecoder_1C.implementation=GPS_L1_CA_Telemetry_Decoder
TelemetryDecoder_1C.dump=false
```

See the Reference Documents for more information about the signal format.

1.6.4 Observables

GNSS systems provide different kinds of observations. The most commonly used are the code observations, also called pseudoranges. The *pseudo* comes from the fact that on the receiver side the clock error is unknown and thus the measurement is not a pure range observation. High accuracy applications also use the carrier phase observations, which are based on measuring the difference between the carrier phase transmitted by the GNSS satellites and the phase of the carrier generated in the receiver. Both observables are computed from the outputs of the tracking module and the decoding of the navigation message. This module collects all the data provided by every tracked channel, aligns all received data into a coherent set, and computes the observables.

The common interface is ObservablesInterface.

Configuration example:

```
;######### OBSERVABLES CONFIG ############
Observables.implementation=Hybrid_Observables

;#dump: Enable or disable the Observables internal binary data file logging [true] or [false]
Observables.dump=false

;#dump_filename: Log path and filename.
Observables.dump_filename=./observables.dat
```

1.6.5 Computation of Position, Velocity and Time

GNSS-SDR provides a module that can compute standalone position fixes and an opt-in NTRIP-corrected RTK solution for dual-band GPS (L1 C/A + L2C or L1 C/A + L5), Galileo (E1 + E5a), and combined GPS+Galileo receivers. It can also store observables and navigation data for use with tools such as GNSSTk or gLAB.

The common interface is PvtInterface. For instance, in order to use the implementation RTKLIB_PVT, add to the configuration file:

```
;######### PVT CONFIG ############
PVT.implementation=RTKLIB_PVT

;#nmea_dump_filename: NMEA log path and filename
PVT.nmea_dump_filename=./gnss_sdr_pvt.nmea;

;#flag_nmea_tty_port: Enable or disable the NMEA log to a serial TTY port (Can be used with real hardware or v
PVT.flag_nmea_tty_port=true;

;#nmea_dump_devname: serial device descriptor for NMEA logging
PVT.nmea_dump_devname=/dev/pts/4

;#dump: Enable or disable the PVT internal binary data file logging [true] or [false]
PVT.dump=false
```

This implementation allows tuning of the following parameters:

```
PVT.implementation=RTKLIB_PVT
PVT.positioning_mode=Single      ; options: Single, Static, Kinematic, PPP_Static, PPP_Kinematic
PVT.satellite_ephemeris=Broadcast ; options: Broadcast, SBAS
PVT.sbas_satellite=0             ; 0: first SBAS GEO received, or select PRN 120-138
PVT.iono_model=Broadcast         ; options: OFF, Broadcast, SBAS, Iono-Free-LC, Estimate_STEC, IONEX
PVT.trop_model=Saastamoinen      ; options: OFF, Saastamoinen, SBAS, Estimate_ZTD, Estimate_ZTD_Grad
PVT.rinex_version=2              ; options: 2 or 3
PVT.output_rate_ms=100           ; Period in [ms] between two PVT outputs
PVT.display_rate_ms=500          ; Position console print (std::out) interval [ms].
PVT.nmea_dump_filename=./gnss_sdr_pvt.nmea ; NMEA log path and filename
PVT.flag_nmea_tty_port=false     ; Enables the NMEA log to a serial TTY port
PVT.nmea_dump_devname=/dev/pts/4 ; serial device descriptor for NMEA logging
PVT.flag_rtcm_server=true        ; Enables an outbound raw TCP server dispatching generated RTCM messages
PVT.ntrip_client_enabled=false   ; Enables inbound RTCM corrections from an NTRIP caster (physical base or VRS
PVT.flag_rtcm_tty_port=false     ; Enables the RTCM log to a serial TTY port
PVT.rtcm_dump_devname=/dev/pts/1 ; serial device descriptor for RTCM logging
PVT.rtcm_tcp_port=2101
PVT.rtcm_MT1019_rate_ms=5000
PVT.rtcm_MT1045_rate_ms=5000
PVT.rtcm_MT1097_rate_ms=1000
PVT.rtcm_MT1077_rate_ms=1000
```

1.6.5.1 Dual-band RTK through NTRIP

The `RTKLIB_PVT` implementation can obtain base-station corrections from an NTRIP caster. This feature is opt-in: `PVT.ntrip_client_enabled` defaults to `false`, and no caster connection is attempted while it is disabled. Per constellation, and freely combined, the receiver may run GPS L1 C/A (`1C`) alone or together with one of L2C (`2S`) or L5 (`L5`), Galileo E1 (`1B`) alone or together with E5a (`5X`), and BeiDou B1C (`1D`, single-frequency); a second band without its first band, both GPS second bands at once, and other signals are rejected. Single-band sets run single-frequency RTK, which is viable on the short effective baselines of VRS services and benefits from combining both constellations. `PVT.navigation_system` must match the enabled constellations exactly ($$ 1 $$ GPS, $$ 8 $$ Galileo, $$ 32 $$ BeiDou, or their sum), and the number of frequency slots is derived from the channel set ($$ 1 $$ for single-band sets; $$ 2 $$ with GPS L2C; $$ 3 $$ whenever L5 or E5a is involved, since both occupy the third RTKLIB slot). GPS L5 and Galileo E5a share the same center frequency, and BeiDou B1C shares the GPS L1 / Galileo E1 center, so the combined GPS L1+L5 / Galileo E1+E5a dual-frequency receiver needs only two RF channels, and a single-frequency GPS+Galileo+BeiDou receiver needs one. Select `Static` for a stationary rover or `Kinematic` for a moving rover; other positioning modes are rejected when the NTRIP client is enabled.

An NTRIP-corrected rover configuration can include:

```
Channels_1C.count=8
Channels_2S.count=8

PVT.implementation=RTKLIB_PVT
PVT.positioning_mode=Kinematic
PVT.num_bands=2
```

```
PVT.navigation_system=1

PVT.ntrip_client_enabled=true
PVT.ntrip_caster_address=caster.example.org
PVT.ntrip_caster_port=2101
PVT.ntrip_mountpoint=EXAMPLE_MOUNTPOINT
PVT.ntrip_username=example_user
PVT.ntrip_password=
PVT.ntrip_password_env=GNSS_SDR_NTRIP_PASSWORD
PVT.ntrip_version=2
PVT.ntrip_tls_enabled=false
PVT.ntrip_inactivity_timeout_ms=10000
PVT.ntrip_reconnect_interval_ms=10000
PVT.ntrip_max_correction_age_s=5.0
PVT.ntrip_station_id=0
PVT.ntrip_fallback_to_single=true
PVT.ntrip_send_gga=true
PVT.ntrip_gga_period_ms=10000
```

The caster address, mountpoint, username, password, and password-environment variable name default to empty strings; the remaining defaults are those shown above, except that `PVT.ntrip_client_enabled` defaults to `false`. Set either `PVT.ntrip_password` or `PVT.ntrip_password_env`, never both. The latter names an environment variable from which GNSS-SDR reads the password and avoids placing the password in the configuration file. Station ID `0` accepts any station; otherwise, set the expected RTCM station ID. Setting `PVT.ntrip_reconnect_interval_ms=0` disables operational reconnect attempts; the one immediate v2-to-v1 compatibility retry still runs and is not counted as an automatic reconnect.

The correction stream must provide an RTCM 3 base position message 1005 or 1006 and observation messages carrying at least each satellite's first band: for GPS, L1 (with L2 or L5 when available) as messages 1002 or 1004 or as MSM4 through MSM7 (1074 through 1077); for Galileo, E1 (with E5a when available) as MSM4 through MSM7 (1094 through 1097); for BeiDou, B1C as MSM4 through MSM7 (1124 through 1127), where the decoder prefers B1C over B1I when the base broadcasts both in the shared first slot. Base satellites providing only the first band contribute single-band double differences; the second band of the receiver's pair is used whenever both sides supply it. Corrections older than `PVT.ntrip_max_correction_age_s` are not used. Enabling this client forces a separate internal single-point solver for receiver-clock correction, independently of the RTK solver. With `PVT.ntrip_fallback_to_single=true`, the receiver reports an explicitly labeled `SOLQ_SINGLE` solution while corrections are unavailable, incomplete, or stale; it does not label that fallback as an RTK solution.

This client sends an NTRIP version 2 HTTP/1.1 request by default and accepts a legacy v1 reply on that connection. If the fully-sent v2 exchange closes or times out before receiving any response bytes, or returns HTTP 400, 501, or 505, it opens one fresh connection and retries using the v1 HTTP/1.0 request. Once selected, v1 is retained for operational reconnects until the client is stopped and started again. DNS, TCP, TLS, and partial-write failures before the request is sent do not trigger a downgrade; `PVT.ntrip_version=1` forces v1 from the first connection. Setting `PVT.ntrip_tls_enabled=true` encrypts both attempts with TLS 1.2 or newer, validating the caster certificate against the system CA store and its host name. Without TLS, Basic authentication only encodes credentials; it does not encrypt them, so use a trusted network or a suitable secure tunnel, and prefer `PVT.ntrip_password_env` over a password stored in a file. VRS and nearest-station mountpoints that require rover GGA reports are supported: with `PVT.ntrip_send_gga=true` (the default), the client periodically sends an NMEA GGA sentence derived from the receiver's current position estimate, every `PVT.ntrip_gga_period_ms` (10 s by default, accepted range 1 s to 24 hours). No GGA is sent before the receiver produces its first position solution; the single-point bootstrap fix is enough for the caster to start serving the virtual station. Set `PVT.ntrip_send_gga=false` if the rover position must not be reported to the caster; physical-base mountpoints work either way.

`PVT.flag_rtcm_server` is unrelated to the NTRIP correction client. It controls an outbound raw TCP server for RTCM messages generated by GNSS-SDR; it neither connects to a caster nor supplies corrections to the RTK solver. The outbound server and inbound NTRIP client can be enabled independently.

For legacy SBAS/EGNOS L1 positioning, use GPS L1 observation channels together with at least one SBAS L1 telemetry channel, and set `PVT.satellite_ephemeris=SBAS`, `PVT.iono_model=SBAS`, and `PVT.trop_model=SBAS`. EGNOS currently augments GPS L1; RTKLIB excludes a satellite when the selected SBAS stream has no valid correction for it. This path provides non-safety-critical Open Service positioning and does not compute protection levels or implement a Safety-of-Life receiver.

1.7 About the software license

GNSS-SDR is released under the General Public License (GPL) v3, thus securing practical usability, inspection, and continuous improvement by the research community, allowing the discussion based on tangible code and the analysis of results obtained with real signals. The GPL implies that:

- Copies may be distributed free of charge or for money, but the source code has to be shipped or provided free of charge (or at cost price) on demand. The receiver of the source code has the same rights meaning he can share copies free of charge or resell.
- The licensed material may be analyzed or modified.
- Modified material may be distributed under the same licensing terms but **do not** have to be distributed.

That means that modifications only have to be made available to the public if distribution happens. So it is perfectly fine to take the GNSS-SDR source code, modify it heavily and use it in a not distributed application / library. This is how companies like Google can run their own patched versions of Linux for example.

But what this also means is that non-GPL code cannot use GPL code. This means that you cannot modify / use GNSS-SDR, blend it with non-GPL code, and make money with the resulting software. You cannot distribute the resulting software under a non-disclosure agreement or contract. Distributors under the GPL also grant a license for any of their patents practiced by the software, to practice those patents in GPL software. You can sell a device that runs with GNSS-SDR, but if you distribute the code, it has to remain under GPL.

1.8 Publications and Credits

If you use GNSS-SDR to produce a research paper or Thesis, we would appreciate if you reference any of these articles to credit the GNSS-SDR project:

- C. Fernández-Prades, J. Arribas, L. Esteve, D. Pubill, P. Closas, *An Open Source Galileo E1 Software Receiver*, in Proc. of the 6th ESA Workshop on Satellite Navigation Technologies (NAVITEC 2012), ESTEC, Noordwijk, The Netherlands, Dec. 2012.
- J. Arribas, *GNSS Array-based Acquisition: Theory and Implementation*, PhD Thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, June 2012.
- C. Fernández-Prades, J. Arribas, P. Closas, C. Avilés, and L. Esteve, *GNSS-SDR: an open source tool for researchers and developers*, in Proc. of the ION GNSS 2011 Conference, Portland, Oregon, Sept. 19-23, 2011.
- C. Fernández-Prades, C. Avilés, L. Esteve, J. Arribas, and P. Closas, *Design patterns for GNSS software receivers*, in Proc. of the 5th ESA Workshop on Satellite Navigation Technologies (NAVITEC'2010), ESTEC, Noordwijk, The Netherlands, Dec. 2010. DOI:10.1109/NAVITEC.2010.5707981

For LaTeX users, these are the BibTeX cites for your convenience:

```
@INPROCEEDINGS{GNSS-SDR12
  author = {C.~{Fern\'{a}ndez--Prades} and J.~Arribas and L.~Esteve and D.~Pubill and P.~Closas},
  title = {An Open Source {G}alileo {E1} Software Receiver},
  booktitle = {Proc. of the 6th ESA Workshop on Satellite Navigation Technologies (NAVITEC'2012)},
  year = {2012},
  address = {ESTEC, Noordwijk, The Netherlands},
  month = {Dec.} }
```

```
@PHDTHESIS{Arribas12,
  author = {J.~Arribas},
  title = {{GNSS} Array-based Acquisition: Theory and Implementation},
  school = {Universitat Polit\'{e}cnica de Catalunya},
  year = {2012},
  address = {Barcelona, Spain},
  month = {June} }


@INPROCEEDINGS{GNSS-SDR11,
 AUTHOR = {C.~{Fern\'{a}ndez--Prades} and J.~Arribas and P.~Closas and C.~Avil\'{e}s and L.~Esteve},
 TITLE = {{GNSS-SDR}: An Open Source Tool For Researchers and Developers},
 BOOKTITLE = {Proc. of the ION GNSS 2011 Conference},
 YEAR = {2011},
 address = {Portland, Oregon},
 month = {Sept.} }


@INPROCEEDINGS{GNSS-SDR10,
 AUTHOR = {C.~{Fern\'{a}ndez--Prades} and C.~Avil\'{e}s and L.~Esteve and J.~Arribas and P.~Closas},
 TITLE = {Design patterns for {GNSS} software receivers},
 BOOKTITLE = {Proc. of the 5th ESA Workshop on Satellite Navigation Technologies (NAVITEC'2010)},
 YEAR = {2010},
 address = {ESTEC, Noordwijk, The Netherlands},
 month = {Dec.},
 note = {doi:10.1109/NAVITEC.2010.5707981} }
```

More papers related to GNSS-SDR are available at the publications page.

1.9 Ok, now what?

In order to start using GNSS-SDR, you may want to populate `gnss-sdr/data` folder (or anywhere else on your system) with raw data files. By "raw data" we mean the output of a Radio Frequency front-end's Analog-to-↩ Digital converter. GNSS-SDR needs signal samples already in baseband or in passband, at a suitable intermediate frequency (on the order of MHz). Prepare your configuration file, and then you are ready for going to the `gnss-sdr/install` folder, running `./gnss-sdr`, and see how the file is processed. Please ask the Developer Team for a signal sample if you need one, and they will do their best ;-)

Another interesting option is working in real-time with a RF front-end. We provide drivers for UHD-compatible hardware (see Signal Source) and for some DVB-T USB dongles. Start with a low number of channels and then increase it in order to test how many channels your processor can handle in real-time.

You can find more information at the **GNSS-SDR Documentation page** or directly asking to the **GNSS-SDR Developers mailing list**.

You are also very welcome to contribute to the project, there are many ways to **participate in GNSS-SDR**. If you need some special feature not yet implemented, the Developer Team would love to be hired for developing it. Please do not hesitate to **contact them**.

Enjoy GNSS-SDR!

The Developer Team.

Chapter 2

Reference Documents

2.1 Interface Control Documents

2.1.1 GPS

All the current GPS Interface Control Documents can be downloaded from GPS.gov, the official U.S. Government webpage for GPS.

- GPS L1 and L2C: Global Positioning System Directorate, **Interface Specification IS-GPS-200 Revision M**. May, 2021.
- GPS L1C (available with first Block III launch): Global Positioning System Directorate, **Interface Specification IS-GPS-800 Revision H**. May, 2021.
- GPS L5 (first Block IIF satellite launched on May, 2010): Global Positioning System Directorate, **Interface Specification IS-GPS-705 Revision H**. May, 2021.

2.1.2 GLONASS

Official GLONASS webpage: Information-analytical centre official website.

- Standard Accuracy (ST) signals at L1 and L2: Russian Institute of Space Device Engineering, Global Navigation Satellite System GLONASS. **Interface Control Document. Navigational radiosignal in bands L1, L2. Edition 5.1**, Moscow, Russia, 2008
- **GLONASS Interface Control Document. Open CDMA navigational radio signal in L1 band. Edition 1.0 (in Russian)**. Russian Space Systems OJSC. 2016.
- **GLONASS Interface Control Document. Open CDMA navigational radio signal in L2 band. Edition 1.0 (in Russian)**. Russian Space Systems OJSC. 2016.
- **GLONASS Interface Control Document. Open CDMA navigational radio signal in L3 band. Edition 1.0 (in Russian)**. Russian Space Systems OJSC. 2016.

2.1.3 Galileo

Check the Galileo website of the European Global Navigation Satellite Systems Agency (GSA) and the Galileo website of the European Space Agency. There is a website with Galileo constellation status information from the GSA.

- Galileo E5, E6, and E1: European GNSS (Galileo) Open Service. **Signal In Space Interface Control Document. Ref: OS SIS ICD, Issue 2.0**, European Union, Jan. 2021.

The European Commission is granting free access to the technical information on the future Galileo open service signal, i.e. the specifications manufacturers and developers need to process data received from satellites. This document informs receiver manufacturers, application developers and service providers on how to use the future Galileo system and what they can expect in terms of performance.

2.1.4 BeiDou

Official webpage at beidou.gov.cn

- **BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0)**. China Satellite Navigation Office, Feb. 2019.

- **BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1C (Version 1.0)**. China Satellite Navigation Office, Jun. 2018.

- **BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B3I (Version 1.0)**. China Satellite Navigation Office, Feb. 2018.

- **BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B2a (Version 1.0)**. China Satellite Navigation Office, Dec. 2017.

- **BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal (Version 2.1)**. China Satellite Navigation Office, December 2016.

2.1.5 Satellite Based Augmentation Systems (SBAS)

- **Minimum Operational Performance Standards for Global Positioning System/Wide Area Augmentation System Airborne Equipment, DO-229D**, RTCA, Washington, DC, Dec. 13, 2006. The 'RTCA MOPS DO229D - appendix A' is the reference standard for WAAS/EGNOS application development. RTCA is an advisory committee of the US federal government, and issues standards for civil airborne equipment, among other duties. One such standard is MOPS 229D (Minimum Operational Performance Standards for Global Positioning System/Wide Area Augmentation System Airborne Equipment version D), which describes the implementation of satellite-based augmentation services (SBAS) for receivers designed for civil aviation use. An annex to DO229D contains the specifications for the SBAS signal and message. The RTCA provides regular updates to these standards. MOPS 229D is available for a fee from the RTCA website.

- **Global Positioning System Wide Area Augmentation System (WAAS) Performance Standard, 1st Edition**, Department of Transportation and Federal Aviation Administration, Oct. 31, 2008. This document defines the levels of performance the U.S. Government makes available to users of the GPS SPS augmented by the Wide Area Augmentation System.

- **EGNOS Data Access Service (EDAS) Service Definition Document. Revision 2.2**, European GNSS Agency (GSA), June, 2019. This is a complementary document to the RTCA DO229D, mentioned above. It describes the scope of services provided by the EGNOS EDAS Service to be used by end-users or Application Specific Service Providers. It details the general conditions relating to the use of the EGNOS service, a technical description of the Signal-in-Space (SIS), the reference receiver, environmental conditions, the service performance achieved and aspects relating to service provision.

- **EGNOS Safety of Life Service Definition Document. Revision 3.3**, European GNSS Agency (GSA), Mar, 2019. The EGNOS Safety of Life (SoL) Service is provided openly and is freely accessible without any direct charge and is tailored to safety-critical transport applications in various domains, in particular for aviation applications. The service is thus compliant with the aviation APV-I (Approach with Vertical Guidance) requirements, as defined by ICAO in Annex 10, but may support also applications in other SoL domains.

- **EGNOS Open Service Service Definition Document. Revision 2.3**, European GNSS Agency (GSA), Sep., 2017.

More information about EGNOS can be found through the EGNOS Portal.

2.2 Other Standards

2.2.1 RINEX

The final output of a navigation receiver is usually its position, speed or other related physical quantities. However, the calculation of those quantities are based on a series of measurements from one or more satellite constellations. Although receivers calculate positions in real time, in many cases it is interesting to store intermediate measures for later post-processing. RINEX is the standard format that allows the management and disposal of the measures generated by a receiver, as well as their off-line processing by a multitude of applications.

- The most common version at present is **RINEX: The Receiver Independent Exchange Format Version 2.12**, which enables storage of measurements from pseudorange, carrier-phase and Doppler systems for GPS, GLONASS, Galileo along with data from EGNOS and WAAS satellite based augmentation systems (SBAS).

- The most recent version is **RINEX: The Receiver Independent Exchange Format Version 3.03** published in July, 2015. It includes Galileo and improves the handling of multi-constellation data files.

- There is also available the **RINEX Extensions to Handle Clock Information**, published in September, 2010.

2.2.2 NMEA

The National Marine Electronics Association released the NMEA 0183 Interface Standard, which defines electrical signal requirements, data transmission protocol and time, and specific sentence formats for a 4800-baud serial data bus. The standard is available for purchase.

2.2.3 KML

KML is an XML language focused on geographic visualization, including annotation of maps and images. Geographic visualization includes not only the presentation of graphical data on the globe, but also the control of the user's navigation in the sense of where to go and where to look. Google submitted KML (formerly Keyhole Markup Language) to the Open Geospatial Consortium (OGC) to be evolved within the OGC consensus process with the following goal: KML Version 2.2 has been adopted as an OGC implementation standard.

- Open Geospatial Consortium, Inc., **OGC KML Version 2.2.0**, April 2008.

2.2.4 C++ Standards

The C++ programming language is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in December 2017 as ISO/IEC 14882:2017 (informally known as C++17). The list of supported C++ standards (the highest available is automatically selected by the CMake script):

- **Draft C++23**: Check the C++ standard draft sources at GitHub.
- **C++20**: The current ISO C++ standard is officially known as *ISO International Standard ISO/IEC 14882:2020(E) – Programming languages – C++*. You can get it from ISO, IEC or ANSI. The closest free working document available is N4868.
- **C++17**: A former ISO C++ standard was officially known as *ISO International Standard ISO/IEC 14882:2017 – Programming languages – C++*. You can get it from ISO, IEC or ANSI. The closest free working document available is N4659.
- **C++14**: A former ISO C++ standard was officially known as *ISO International Standard ISO/IEC 14882:2014 – Programming languages – C++*. You can get it from ISO or ANSI. The closest free working document available is N4296.
- **C++11**: An older ISO C++ standard was ISO/IEC 14882:2011. You can get it from ISO. The closest free working document available is N3337.

2.2.5 Positioning protocols in wireless communication networks

Cellular industry location standards first appeared in the late 1990s, with the 3rd generation partnership project (3GPP) radio resource location services protocol (RRLP) technical specification 44.031 positioning protocol for GSM networks. Today, RRLP is the de facto standardized protocol to carry GNSS assistance data to GNSS-enabled mobile devices, and the term "3GPP specification" now covers all GSM (including GPRS and EDGE), W-CDMA and LTE (including LTE-A) specifications. Precisely, the label "LTE-A" is applied to networks compliant with LTE Release 10 and beyond, which fulfill the requirements issued by the International Telecommunication Union Radiocommunication Sector (ITU-R) in the global standard for international mobile telecommunications (IMT Advanced, also referred to as 4G) access technologies. Control plane protocols: \li Radio Resource LCS Protocol (RRLP): <b>3GPP

Technical Specification 44.031. \li LTE Positioning Protocol (LPP): <a href="https://www.↩
3gpp.org/ftp/Specs/html-info/36355.htm" target="_blank">3GPP Technical Specification 36.355. User plane protocols: \li Open Mobile Alliance (OMA), <a href="http://member.↩
openmobilealliance.org/ftp/Public_documents/LOC/Permanent_documents/OMA-↩
AD-SUPL-V1_0-20070615-A.zip" target="_blank">Secure User Plane Location Architecture Version 1 (SUPL 1.0), June 2007. \li Open Mobile Alliance (OMA), <a href="http://member.↩
openmobilealliance.org/ftp/Public_documents/LOC/Permanent_documents/OMA-↩
AD-SUPL-V2_0-20120417-A.zip" target="_blank">Secure User Plane Location Architecture Version 2 (SUPL 2.0), April 2012. LTE Release 9 introduced extension hooks in LPP messages, so that the bodies external to 3GPP could extend the LPP feature set. OMA LPP extensions (LPPe), supported in SUPL 3.0, build on top of the 3GPP LPP reusing its procedures and data types. Check the <a href="https://technical.↩
openmobilealliance.org/index.html" target="_blank">OMA Specifications webpage for updated information about LPP Extensions (LPPe) Specification. \li The <a href="http://member.↩
openmobilealliance.org/ftp/Public_documents/loc/Permanent_documents/OMA-↩
TS-MLP-V3_5-20181119-D.zip" target="_blank">**OMA Mobile Location Protocol (MLP) V3.5** is an application-level protocol for getting the position of mobile stations (mobile phones, wireless personal digital assistants, etc.) independent of underlying network technology. The MLP serves as the interface between a Location Server and a Location Services (LCS) Client. This specification defines the core set of operations that a Location Server should be able to perform.

Chapter 3

Signal model

3.1 GNSS signal model

This page describes signals transmitted by GNSS space vehicles. Signal models are mathematical representations of the electromagnetic waves that are exciting the receiver's antenna. The current induced by those waves is then amplified, filtered and downconverted to a suitable frequency (can be at some intermediate frequency or directly to baseband), and then converted to 0s and 1s by the Analog-to-Digital Converter (ADC). That is the job of the Radio Frequency front-end, which at its output delivers a stream of digital samples. Those samples constitute the input of a software receiver, so for GNSS-SDR the signal models described below can be seen as *the rules of the game*.

GNSS' space vehicles are modern versions of lighthouses, but with better visibility. Each satellite is a reference point, and if we know our distance to several reference points, we can compute our location, just as mariners do when they see a couple of lighthouses. For each in-view satellite i of system s, we can write:

$$\rho_i = \sqrt{\left(x_i^{\text{Tx}} - x\right)^2 + \left(y_i^{\text{Tx}} - y\right)^2 + \left(z_i^{\text{Tx}} - z\right)^2} + c\Delta t^{(s)} + \sigma_e, \tag{3.1}$$

where $\left(x_i^{\text{Tx}}, y_i^{\text{Tx}}, z_i^{\text{Tx}}\right)$ is the satellite's position (known from the navigation message), (x, y, z) the receiver's position, and σ_e gathers other sources of error. Since the receiver needs to estimate its own 3D position (three spatial unknowns) and its clock deviation with respect to the satellites' time basis, at least $3 + N_s$ satellites must be seen by the receiver at the same time, where N_s is the number of different navigation systems available (in-view) at a given time. Each received satellite signal, once synchronized and demodulated at the receiver, defines one equation such as the one defined above, forming a set of nonlinear equations that can be solved algebraically by means of the `Bancroft algorithm` or numerically, resorting to multidimensional Newton-Raphson and weighted least square methods. When *a priori* information is added we resort to Bayesian estimation, a problem that can be solved recursively by a Kalman filter or any of its variants. The problem can be further expanded by adding other unknowns (for instance, parameters of ionospheric and tropospheric models), sources of information from other systems, mapping information, and even motion models of the receiver. In the design of multi-constellation GNSS receivers, the vector of unknowns can also include the receiver clock offset with respect to each system in order to take advantage of a higher number of in-view satellites and using them jointly in the navigation solution, therefore increasing accuracy.

The `analytic representation` of a signal received from a GNSS satellite can be generically expressed as

$$r(t) = \alpha(t) s_T\left(t - \tau(t)\right) e^{-j2\pi f_d(t)} e^{j2\pi f_c t} + n(t)\ , \tag{3.2}$$

where $\alpha(t)$ is the amplitude, $s_T(t)$ is the complex baseband transmitted signal, $\tau(t)$ is the time-varying delay, $f_d(t) = f_c\tau(t)$ is the Doppler shift, f_c is the carrier frequency, and $n(t)$ is a noise term. These signals arrive to the Earth's surface at extremely low power (e.g. -158.5 dBW for GPS L1 C/A-code, -157 dBW for Galileo E1), well below the noise floor. In order to estimate its distances to satellites, the receiver must correlate time-aligned replicas of the corresponding pseudorandom code with the incoming signal, in a process called *despreading* that

provides processing gain only to the signal of interest. After a coarse and fine estimation stages of the synchronization parameters (usually known as acquisition and tracking, respectively), signal processing output is in form of *observables*:

i) the pseudorange (code) measurement, equivalent to the difference of the time of reception (expressed in the time frame of the receiver) and the time of transmission (expressed in the time frame of the satellite) of a distinct satellite signal; and optionally

ii) the carrier-phase measurement, actually being a measurement on the beat frequency between the received carrier of the satellite signal and a receiver-generated reference frequency. Carrier phase measurements are ambiguous, in the sense that the integer number of carrier wavelengths between satellite and the receiver's antenna is unknown. Techniques such as Least-square AMBiguity Decorrelation Approach (LAMBDA) or Multi Carrier Ambiguity Resolution (MCAR) can be applied to resolve such ambiguity and provide an accurate estimation of the distance between the satellite and the receiver.

Then, depending on the required accuracy, the navigation solution can range from pseudorange-only, computationally low demanding, and limited accuracy least squares methods to sophisticated combinations of code and phase observables at different frequencies for high demanding applications such as surveying, geodesy, and geophysics.

Next sections provide brief descriptions of the space segment of different GNSSs and their broadcast signal structures accessible by civilians.

3.1.1 Global Positioning System (GPS) signal in space

The Global Positioning System (GPS) is a space-based radio-navigation system owned by the United States Government (USG) and operated by the United States Air Force (USAF). GPS provides positioning and timing services to military and civilian users on a continuous, worldwide basis. Two GPS services are provided: the Precise Positioning Service (PPS), available primarily to the military of the United States and its allies, and the Standard Positioning Service (SPS) open to civilian users.

- **GPS L1**. Defined at **Interface Specification IS-GPS-200 Revision M**, this band is centered at $f_{\text{GPS L1}} = 1575.42$ MHz. The complex baseband transmitted signal can be written as

$$s_T^{(\text{GPS L1})}(t) = e_{L1I}(t) + j e_{L1Q}(t) \ , \tag{3.3}$$

with

$$e_{L1I}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\Big[[l]_{204600}\Big] \oplus C_{\text{P(Y)}}\Big[|l|_{L_{\text{P(Y)}}}\Big] p(t - lT_{c,\text{P(Y)}}) \ , \tag{3.4}$$

$$e_{L1Q}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\Big[[l]_{20460}\Big] \oplus C_{\text{C/A}}\Big[|l|_{1023}\Big] p(t - lT_{c,\text{C/A}}) \ , \tag{3.5}$$

where $\oplus$ is the exclusive-or operation (modulo-2 addition), $|l|_L$ means l modulo L, $[l]_L$ means the integer part of $\frac{l}{L}$, D_{NAV} is the GPS navigation message bit sequence, transmitted at 50 bps, $T_{c,\text{P(Y)}} = \frac{1}{10.23}$ μs, $T_{c,\text{C/A}} = \frac{1}{1.023}$ μs, $L_{\text{P(Y)}} = 6.1871 \cdot 10^{12}$, and $p(t)$ is a rectangular pulse of a chip-period duration centered at $t = 0$ and filtered at the transmitter. According to the chip rate, the binary phase-shift keying modulations in the equations above are denoted as BPSK(10) and BPSK(1), respectively. The precision P codes (named Y codes whenever the anti-spoofing mode is activated, encrypting the code and thus denying non-↩ U.S. military users) are sequences of 7 days in length. Regarding the modernization plans for GPS, it is worthwhile to mention that there is a new civilian-use signal planned, called L1C and defined at **Interface Specification IS-GPS-800 Revision H**, to be broadcast on the same L1 frequency that currently contains the C/A signal. The L1C is available with first Block III launch. The implementation provides C/A code to ensure backward compatibility.

- **GPS L2C**. Defined at **Interface Specification IS-GPS-200 Revision M**, is only available on Block IIR-M and subsequent satellite blocks. Centered at $f_{\text{GPS L2}} = 1227.60$ MHz, the signal structure is the same than in (eq:GPSL1}), with the precision code in the In-phase component, just as in (eq:L1CAI}) but with an optional presence of the navigation message D_{NAV}. For the Quadrature-phase component, three options are defined:

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} D_{\text{CNAV}}\left[[l]_{10230}\right] \oplus \left(C_{\text{CL}}\left[|l|_{L_{\text{CL}}}\right] p_{1/2}\left(t - lT_{c,L2C}\right) + \right. \tag{3.6}$$

$$\left. + C_{\text{CM}}\left[|l|_{L_{\text{CM}}}\right] p_{1/2}\left(t - \left(l + \frac{3}{4}\right) T_{c,L2C}\right)\right), \tag{3.7}$$

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\left[[l]_{20460}\right] \oplus C_{\text{C/A}}\left[|l|_{1023}\right] p\left(t - lT_{c,\text{C/A}}\right), \text{ or} \tag{3.8}$$

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} C_{\text{C/A}}\left[|l|_{1023}\right] p(t - lT_{c,\text{C/A}}), \tag{3.9}$$

where $T_{c,L2C} = \frac{1}{511.5}$ ms and $p_{1/2}(t)$ is a rectangular pulse of half chip-period duration, thus time-multiplexing both codes. The civilian long code C_{CL} is $L_{\text{CL}} = 767250$ chips long, repeating every 1.5 s, while the civilian moderate code C_{CM} is $L_{\text{CL}} = 10230$ chips long and its repeats every 20 ms. The CNAV data is an upgraded version of the original NAV navigation message, containing higher precision representation and nominally more accurate data than the NAV data. It is transmitted at 25 bps with forward error correction (FEC) encoding, resulting in 50 sps.

- **GPS L5**. The GPS L5 link, defined at **Interface Specification IS-GPS-705 Revision H**, is only available in Block IIF (first satellite launched on May, 2010) and subsequent satellite blocks. Centered at $f_{\text{GPS L5}} = 1176.45$ MHz, this signal in space can be written as:

$$s_T^{(\text{GPS L5})}(t) = e_{L5I}(t) + je_{L5Q}(t), \tag{3.10}$$

$$e_{L5I}(t) = \sum_{m=-\infty}^{+\infty} C_{nh_{10}}\left[|m|_{10}\right] \oplus D_{\text{CNAV}}\left[[m]_{10}\right] \oplus \tag{3.11}$$

$$\oplus \sum_{l=1}^{102300} C_{L5I}\left[|l|_{10230}\right] p(t - mT_{c,nh} - lT_{c,L5}),$$

$$e_{L5Q}(t) = \sum_{m=-\infty}^{+\infty} C_{nh_{20}}\left[|m|_{20}\right] \oplus \sum_{l=1}^{102300} C_{L5Q}\left[|l|_{10230}\right] \cdot$$

$$\cdot\, p(t - mT_{c,nh} - lT_{c,L5}), \tag{3.12}$$

where $T_{c,nh} = 1$ ms and $T_{c,L5} = \frac{1}{10.23}$ μs, thus defining a BPSK(10) modulation. Both L5I and L5Q contain synchronization sequences. \end{itemize}

3.1.2 GLONASS signal in space

The nominal baseline constellation of the Russian Federation's Global Navigation Satellite System (GLONASS) comprises 24 GLONASS-M satellites that are uniformly deployed in three roughly circular orbital planes at an inclination of 64.8^o to the equator. The altitude of the orbit is $19,100$ km. The orbit period of each satellite is 11 hours, 15 minutes, and 45 seconds. The orbital planes are separated by 120^o right ascension of the ascending node. Eight satellites are equally spaced in each plane with 45^o argument of latitude. Moreover, the orbital planes have an argument of latitude displacement of 15^o relative to each other.

GLONASS civil signal-in-space is defined at **Interface Control Document. Navigational radiosignal in bands L1, L2. Edition 5.1**. This system makes use of a frequency-division multiple access (FDMA) signal structure, transmitting in two bands: $f_{GLOL1}^{(k)} = 1602 + k \cdot 0.5625$ MHz and $f_{GLOL2}^{(k)} = 1246 + k \cdot 0.4375$ MHz, where $k \in \{-7, -6, \cdots, 5, 6\}$ is the channel number. Satellites in opposite points of an orbit plane transmit signals on equal frequencies, as these satellites will never be in view simultaneously by a ground-based user.

- **GLONASS L1**. Two kind of signals are transmitted: a standard precision (SP) and an obfuscated high precision (HP) signal. The complex baseband transmitted signal can be written as

$$s_T^{(\text{GLO L1})}(t) = e_{L1I}(t) + j e_{L1Q}(t) \, , \tag{3.13}$$

with BPSK(5) and BPSK(0.5) modulations:

$$e_{L1I}(t) = \sum_{l=-\infty}^{\infty} D_{\text{GNAV}}\Big[[l]_{102200}\Big] \oplus C_{\text{HP}}\Big[|l|_{L_{\text{HP}}}\Big] p(t - lT_{c,\text{HP}}) \, , \tag{3.14}$$

$$e_{L1Q}(t) = \sum_{l=-\infty}^{\infty} D_{\text{GNAV}}\Big[[l]_{10220}\Big] \oplus C_{\text{SP}}\Big[|l|_{511}\Big] p(t - lT_{c,\text{SP}}) \, , \tag{3.15}$$

where $T_{c,\text{HP}} = \frac{1}{5.11}$ μs, $T_{c,\text{SP}} = \frac{1}{0.511}$ μs, and $L_{\text{HP}} = 3.3554 \cdot 10^7$. The navigation message D_{GNAV} is transmitted at 50 bps. Details of its content and structure, as well as the generation of the C_{SP} code, can be found at the ICD. The usage of the HP signal should be agreed with the Russian Federation Defense Ministry, and no more details have been disclosed.

- **GLONASS L2**. Beginning with the second generation of satellites, called GLONASS-M and first launched in 2001, a second civil signal is available using the same SP code than the one in the L1 band.

The use of FDMA techniques, in which the same code is used to broadcast navigation signals on different frequencies, and the placement of civil GLONASS transmissions on frequencies close to 1600 MHz, well above the GPS L1 band, have complicated the design of combined GLONASS/GPS receivers, particularly low-cost equipment for mass-market applications. Future plans of modernization are intended to increase compatibility and interoperability with other GNSS, and include the addition of a code-division multiple access (CDMA) structure, and possibly binary offset carrier (BOC) modulation, beginning with the third civil signal in the L3 band ($1197.648 - 1212.255$ MHz). Russia is implementing the new signals on the next-generation GLONASS-K satellites, with a first prototype successfully launched into orbit on February 26, 2011.

3.1.3 Galileo signal in space

The nominal Galileo constellation comprises a total of 27 operational satellites (plus 3 active spares), that are evenly distributed among three orbital planes inclined at 56^o relative to the equator. There are nine operational satellites per orbital plane, occupying evenly distributed orbital slots. Three additional spare satellites (one per orbital plane) complement the nominal constellation configuration. The Galileo satellites are placed in quasi-circular Earth orbits with a nominal semi-major axis of about $30,000$ km and an approximate revolution period of 14 hours. The Control segment full infrastructure will be composed of $30 - 40$ sensor stations, 3 control centers, 9 Mission Uplink stations, and 5 TT&C stations.

Galileo's Open Service is defined at **Signal In Space Interface Control Document. Ref: OS SIS ICD, Issue 2.0**, where the following signal structures are specified:

- **Galileo E1**. This band, centered at $f_{\text{Gal E1}} = 1575.420$ MHz and with a reference bandwidth of 24.5520 MHz, uses the so-called composite binary offset carrier CBOC(6,1, $\frac{1}{11}$) modulation, defined in baseband as:

$$s_T^{(\text{Gal E1})}(t) = \frac{1}{\sqrt{2}}\Big(e_{E1B}(t) \left(\alpha sc_A(t) + \beta sc_B(t)\right) + \tag{3.16}$$

$$- e_{E1C}(t) \left(\alpha sc_A(t) - \beta sc_B(t)\right) \Big) \, , \tag{3.17}$$

where the subcarriers $sc(t)$ are defined as

$$sc_A(t) = \text{sign}\Big(\sin(2\pi f_{s,E1A} t) \Big) \, , \tag{3.18}$$

$$sc_B(t) = \text{sign}\Big(\sin(2\pi f_{s,E1B} t) \Big) \, , \tag{3.19}$$

and $f_{s,E1A} = 1.023$ MHz, $f_{s,E1B} = 6.138$ MHz are the subcarrier rates, $\alpha = \sqrt{\frac{10}{11}}$, and $\beta = \sqrt{\frac{1}{11}}$. Channel B contains the I/NAV type of navigation message, $D_{\text{I/NAV}}$, intended for Safety-of-Life (SoL) services:

$$e_{E1B}(t) = \sum_{l=-\infty}^{+\infty} D_{\text{I/NAV}}\Big[[l]_{4092}\Big] \oplus C_{E1B}\Big[|l|_{4092}\Big] p(t - lT_{c,E1B}). \tag{3.20}$$

In case of channel C, it is a pilot (dataless) channel with a secondary code, forming a tiered code:

$$\begin{aligned} e_{E1C}(t) = &\sum_{m=-\infty}^{+\infty} C_{E1Cs}\Big[|m|_{25}\Big] \oplus \sum_{l=1}^{4092} C_{E1Cp}\Big[l\Big] \cdot \\ &\cdot p(t - mT_{c,E1Cs} - lT_{c,E1Cp}) \ , \end{aligned} \tag{3.21}$$

with $T_{c,E1B} = T_{c,E1Cp} = \frac{1}{1.023}$ μs and $T_{c,E1Cs} = 4$ ms. The C_{E1B} and C_{E1Cp} primary codes are pseudorandom memory code sequences defined at Annex C of OS SIS ICD. The binary sequence of the secondary code C_{E1Cs} is 00111000000010101101100 10. This band also contains another component, Galileo E1A, intended for the Public Regulated Service (PRS). It uses a BOC(15,2.5) modulation with cosine-shaped subcarrier $f_{s,E1A} = 15.345$ MHz and $T_{c,E1A} = \frac{1}{2.5575}$ μs. The PRS spreading codes and the structure of the navigation message have not been made public.

- **Galileo E6**. Intended for the Commercial Service and centered at $f_{\text{Gal E6}} = 1278.750$ MHz, this band provides pilot and data components

$$s_T^{(\text{Gal E6})}(t) = \frac{1}{\sqrt{2}} \left(e_{E6B}(t) - e_{E6C}(t)\right) \ , \tag{3.22}$$

$$\begin{aligned} e_{E6B}(t) = &\sum_{m=-\infty}^{+\infty} D_{\text{C/NAV}}\Big[[l]_{5115}\Big] \oplus C_{E6B}\Big[|l|_{L_{E6B}}\Big] \cdot \\ &\cdot p(t - lT_{c,E6}), \end{aligned} \tag{3.23}$$

$$\begin{aligned} e_{E6C}(t) = &\sum_{m=-\infty}^{+\infty} C_{E6Cs}\Big[|m|_{100}\Big] \oplus \sum_{l=1}^{L_{E6C}} C_{E6Cp}\Big[l\Big] \cdot \\ &\cdot p(t - mT_{c,E6s} - lT_{c,E6p}), \end{aligned} \tag{3.24}$$

where $D_{\text{C/NAV}}$ is the C/NAV navigation data stream, which is modulated with the encrypted ranging code C_{E6B} with chip period $T_{c,E6} = \frac{1}{5.115}$ μs, thus being a BPSK(5) modulation. Codes C_{E6B} and primary codes C_{E6Cs} and their respective lengths, L_{E6B} and L_{E6C}, have not been published. The secondary codes for the pilot component, C_{E6Cs}, are available at the OS SIS ICD. The receiver reference bandwidth for this signal is 40.920 MHz. This band also contains another component, Galileo E6A, intended for PRS.

- **Galileo E5**. Centered at $f_{\text{Gal E5}} = 1191.795$ MHz and with a total bandwidth of 51.150 MHz, its signal structure deserves some analysis. The AltBOC modulation can be generically expressed as

$$s^{\text{AltBOC}}(t) = x_1(t)v^*(t) + x_2(t)v(t) \ , \tag{3.25}$$

where $v(t) = \frac{1}{\sqrt{2}} \left(\text{sign}\left(\cos(2\pi f_s t)\right) + j\text{sign}\left(\sin(2\pi f_s t)\right)\right)$ is the single side-band subcarrier, f_s is the subcarrier frequency, $(\cdot)^*$ stands for the conjugate operation, and $x_1(t)$ and $x_2(t)$ are QPSK signals. The resulting waveform does not exhibit constant envelope. In case of Galileo, the need for high efficiency of the satellites' onboard High Power Amplifier (HPA) has pushed a modification on the signal in order to make it envelope-constant and thus use the HPA at saturation. This can be done by adding some inter-modulation products to the expression above, coming up with the following definition:

$$\begin{aligned} s_T^{(\text{Gal E5})}(t) = &e_{E5a}(t)ssc_s^*(t) + e_{E5b}(t)ssc_s(t) + \\ &+ \bar{e}_{E5a}(t)ssc_p^*(t) + \bar{e}_{E5b}(t)ssc_p(t) \ , \end{aligned} \tag{3.26}$$

where the single and product side-band signal subcarriers are

$$ssc_s(t) = sc_s(t) + jsc_s\left(t - \frac{T_s}{4}\right), \tag{3.27}$$

$$ssc_p(t) = sc_p(t) + jsc_p\left(t - \frac{T_s}{4}\right), \tag{3.28}$$

and

$$e_{E5a}(t) = e_{E5aI}(t) + je_{E5aQ}(t), \tag{3.29}$$
$$e_{E5b}(t) = e_{E5bI}(t) + je_{E5bQ}(t), \tag{3.30}$$
$$\bar{e}_{E5a}(t) = \bar{e}_{E5aI}(t) + j\bar{e}_{E5aQ}(t), \tag{3.31}$$
$$\bar{e}_{E5b}(t) = \bar{e}_{E5bI}(t) + j\bar{e}_{E5bQ}(t), \tag{3.32}$$
$$\bar{e}_{E5aI}(t) = e_{E5aQ}(t)e_{E5bI}(t)e_{E5bQ}(t), \tag{3.33}$$
$$\bar{e}_{E5aQ}(t) = e_{E5aI}(t)e_{E5bI}(t)e_{E5bQ}(t), \tag{3.34}$$
$$\bar{e}_{E5bI}(t) = e_{E5bQ}(t)e_{E5aI}(t)e_{E5aQ}(t), \tag{3.35}$$
$$\bar{e}_{E5bQ}(t) = e_{E5bI}(t)e_{E5aI}(t)e_{E5aQ}(t). \tag{3.36}$$

The signal components are defined as

$$e_{E5aI}(t) = \sum_{m=-\infty}^{+\infty} C_{E5aIs}\Big[|m|_{20}\Big] \oplus \sum_{l=1}^{10230} C_{E5aIp}\Big[l\Big] \oplus \tag{3.37}$$
$$\oplus\, D_{\text{F/NAV}}\Big[[l]_{204600}\Big]\, p(t - mT_{c,E5s} - lT_{c,E5p}), \tag{3.38}$$
$$e_{E5aQ}(t) = \sum_{m=-\infty}^{+\infty} C_{E5aQs}\Big[|m|_{100}\Big] \oplus \sum_{l=1}^{10230} C_{E5aQp}\Big[l\Big] \cdot \tag{3.39}$$
$$\cdot\, p(t - mT_{c,E5s} - lT_{c,E5p}), \tag{3.40}$$
$$e_{E5bI}(t) = \sum_{m=-\infty}^{+\infty} C_{E5bIs}\Big[|m|_{4}\Big] \oplus \sum_{l=1}^{10230} C_{E5aIp}\Big[l\Big] \oplus \tag{3.41}$$
$$\oplus\, D_{\text{I/NAV}}\Big[[l]_{40920}\Big]\, p(t - mT_{c,E5s} - lT_{c,E5p}), \tag{3.42}$$
$$e_{E5bQ}(t) = \sum_{m=-\infty}^{+\infty} C_{E5bQs}\Big[|m|_{100}\Big] \oplus \sum_{l=1}^{10230} C_{E5bQp}\Big[l\Big] \cdot \tag{3.43}$$
$$\cdot\, p(t - mT_{c,E5s} - lT_{c,E5p}), \tag{3.44}$$

where $T_{c,E5s} = 1$ ms and $T_{c,E5p} = \frac{1}{10.23}$ μs. Channel A contains the F/NAV type of navigation message, $D_{\text{F/NAV}}$, intended for the Open Service. The I/NAV message structures for the E5bI and E1B signals use the same page layout. Only page sequencing is different, with page swapping between both components in order to allow a fast reception of data by a dual frequency receiver. The single subcarrier $sc_s(t)$ and the product subcarrier $sc_p(t)$ are defined as:

$$sc_s(t) = \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t - \frac{\pi}{4}\right)\right) + \tag{3.45}$$
$$+ \frac{1}{2}\text{sign}\Big(\cos\left(2\pi f_s t\right)\Big) +$$
$$+ \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t + \frac{\pi}{4}\right)\right), \tag{3.46}$$

$$sc_p(t) = -\frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t - \frac{\pi}{4}\right)\right) + \tag{3.47}$$
$$+ \frac{1}{2}\text{sign}\Big(\cos\left(2\pi f_s t\right)\Big) +$$
$$- \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t + \frac{\pi}{4}\right)\right), \tag{3.48}$$

with a subcarrier frequency of $f_s = 15.345$ MHz, thus defining an AltBOC(15,10) modulation. The QPSK(10) signal $e_{E5a}(t)$ defined above is shifted to $f_{\text{Gal E5a}} \doteq f_{\text{Gal E5}} - f_s = 1176.450$ MHz, while $e_{E5b}(t)$ is shifted to $f_{\text{Gal E5b}} \doteq f_{\text{Gal E5}} + f_s = 1207.140$ MHz. Thus, we can bandpass filter around $f_{\text{Gal E5a}}$ and get a good approximation of a QPSK(10) signal, with very low energy components of $e_{E5b}(t)$, $\bar{e}_{E5a}(t)$, and $\bar{e}_{E5b}(t)$:

$$s_T^{(\text{Gal E5a})}(t) \simeq e_{E5aI}(t) + je_{E5aQ}(t). \tag{3.49}$$

The same applies to $e_{E5b}(t)$, allowing an independent reception of two QPSK(10) signals and thus requiring considerably less bandwidth than the processing of the whole E5 band.

3.1.4 Reference

This text is an except of the following paper:

- C. Fernández-Prades, L. Lo Presti, E. Falleti, *Satellite Radiolocalization From GPS to GNSS and Beyond: Novel Technologies and Applications for Civil Mass-Market*. Proceedings of the IEEE. Vol 99, No. 11, pp. 1882-1904. November, 2011. doi: 10.1109/JPROC.2011.2158032

Chapter 4

Todo List

Member ALPHA_0 ({{69, 8}})

read all pages of subframe 4

Member T_OA ({{69, 8}})

read all pages of subframe 5

Chapter 5

Topic Index

5.1 Topics

Here is a list of all topics with brief descriptions:

Acquisition . . . 79
acquisition_adapters . . . 79
acquisition_gr_blocks . . . 80
acquisition_libs . . . 82
Core GNSS Receiver . . . 83
GNSS block interfaces . . . 83
core_libs . . . 84
core_monitor . . . 89
core_receiver . . . 89
core_system_parameters . . . 91
Channel . . . 227
channel_adapters . . . 227
channel_libs . . . 227
Signal Conditioner . . . 228
conditioner_adapters . . . 228
Data Type Adapters . . . 228
data_type_adapters . . . 228
data_type_gr_blocks . . . 229
Input Filter . . . 230
input_filter_adapters . . . 230
input_filter_gr_blocks . . . 230
Algorithms Common Library . . . 231
algorithms_libs . . . 231
gnss_sdr_flags . . . 255
PVT . . . 259
algorithms_libs_rtklib . . . 259
RTKLIB . . . 297
pvt_adapters . . . 297
pvt_gr_blocks . . . 297
pvt_libs . . . 297
Observables . . . 302
obs_adapters . . . 302
obs_gr_blocks . . . 302

observables_libs . . . 303
Resampler . . . 303
resampler_adapters . . . 303
resampler_gr_blocks . . . 303
Signal Source . . . 304
signal_source_adapters . . . 304
signal_source_gr_blocks . . . 306
signal_source_libs . . . 309
Telemetry_Decoder . . . 311
Telemetry_Decoder_adapters . . . 311
telemetry_decoder_gr_blocks . . . 311
telemetry_decoder_libswiftcnav . . . 313
telemetry_decoder_libs . . . 315
Tracking . . . 315
tracking_adapters . . . 315
tracking_gr_blocks . . . 316
tracking_libs . . . 317

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

block_factory
Class that produces all kinds of GNSS blocks 323
gnss_tow 324
std
STL namespace 326

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

sdr_gnss_packet_t::[struct].sats . . . 331
Acq_Conf . . . 331
Acq_Conf_Fpga . . . 336
ad936x_iio_custom . . . 343
ad936x_iio_samples . . . 344
Agnss_Ref_Location . . . 349
Agnss_Ref_Time . . . 350
Allocator . . . 351
alm_t . . . 352
ambc_t . . . 354
An_Packet_Printer . . . 354
an_packet_t . . . 355
Bayesian_estimator . . . 358
Bds3_B1c_AlmanacMedium . . . 359
Bds3_B1c_AlmanacReduced . . . 360
Bds3_B1c_Bgto . . . 361
Bds3_B1c_Eop . . . 362
Bds3_B1c_PageCommon . . . 363
Bds3_B1c_PageData . . . 364
Beidou_Cnav1_Iono . . . 373
Beidou_Cnav1_Navigation_Message . . . 374
Beidou_Cnav1_PageData_Message . . . 375
Beidou_Cnav1_Utc_Model . . . 375
Beidou_Cnav2_Navigation_Message . . . 377
Beidou_Dnav_Differential_Correction . . . 380
Beidou_Dnav_Differential_Corrections . . . 381
Beidou_Dnav_Iono_Grid_Point . . . 390
Beidou_Dnav_Navigation_Message . . . 391
Beidou_Dnav_Utc_Model . . . 395
BeidouLdpcGraph . . . 398
gr::block
  Galileo_E1_Tcp_Connector_Tracking_cc . . . 529
  Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc . . . 670
  Gps_L1_Ca_Gaussian_Tracking_cc . . . 671
  Gps_L1_Ca_Tcp_Connector_Tracking_cc . . . 672

Notch . . . 768
NotchLite . . . 771
acquisition_impl_interface . . . 339
galileo_e5a_noncoherentIQ_acquisition_caf_cc . . . 530
galileo_pcps_8ms_acquisition_cc . . . 561
pcps_acquisition . . . 808
pcps_acquisition_fine_doppler_cc . . . 811
pcps_assisted_acquisition_cc . . . 816
pcps_cccwsr_acquisition_cc . . . 818
pcps_opencl_acquisition_cc . . . 821
pcps_quicksync_acquisition_cc . . . 823
pcps_tong_acquisition_cc . . . 826
ad936x_iio_source . . . 344
channel_msg_receiver_cc . . . 407
channel_status_msg_receiver . . . 408
direct_resampler_conditioner_cb . . . 440
direct_resampler_conditioner_cc . . . 441
direct_resampler_conditioner_cs . . . 441
dll_pll_veml_tracking . . . 457
dll_pll_veml_tracking_fpga . . . 458
galileo_e6_has_msg_receiver . . . 532
galileo_tow_map . . . 568
gnss_sdr_fpga_sample_counter . . . 621
gnss_sdr_time_counter . . . 628
gnss_synchro_monitor . . . 637
hybrid_observables_gs . . . 704
kf_tracking . . . 729
labsat23_source . . . 730
nav_message_monitor . . . 758
osnma_msg_receiver . . . 798
pulse_blanking_cc . . . 852
signal_generator_c . . . 962
telemetry_impl_interface . . . 997
beidou_b1c_telemetry_decoder_gs . . . 367
beidou_b2a_telemetry_decoder_gs . . . 368
beidou_dnav_telemetry_decoder_gs . . . 393
galileo_telemetry_decoder_gs . . . 567
glonass_gnav_telemetry_decoder_gs . . . 597
gps_l1_ca_telemetry_decoder_gs . . . 672
gps_l2c_telemetry_decoder_gs . . . 674
gps_l5_telemetry_decoder_gs . . . 675
sbas_l1_telemetry_decoder_gs . . . 932
Channel_Event . . . 407
ChannelFsm . . . 409
cl_fft_plan . . . 411
clFFT_Complex . . . 413
clFFT_Dim3 . . . 414
clFFT_SplitComplex . . . 414
cnav_msg_decoder_t . . . 415
cnav_msg_t . . . 415
cnav_v27_part_t . . . 416
Command_Event . . . 418
Concurrent_Map< Data > . . . 420
Concurrent_Queue< Data > . . . 421
HistogramBitSynchronizer::Config . . . 422
ConfigurationInterface . . . 425
FileConfiguration . . . 486
InMemoryConfiguration . . . 712

ControlThread . . . 427
Cpu_Multicorrelator . . . 429
Cpu_Multicorrelator_16sc . . . 430
Cpu_Multicorrelator_Real_Codes . . . 430
CubatureFilter . . . 433
cuda_multicorrelator . . . 434
CudaPcpsEngine . . . 434
Deleter . . . 438
dgps_t . . . 439
Dll_Pll_Conf . . . 444
Dll_Pll_Conf_Fpga . . . 450
DSM_dsm_header . . . 466
DSM_KROOT_message . . . 466
DSM_nma_header . . . 468
DSM_PKR_message . . . 469
eph_t . . . 470
erp_t . . . 475
erpd_t . . . 475
Exponential_Smoother . . . 478
exterr_t . . . 480
fcbd_t . . . 480
file_t . . . 484
Fpga_Acquisition . . . 507
Fpga_buffer_monitor . . . 509
Fpga_DMA . . . 510
Fpga_dynamic_bit_selection . . . 512
Fpga_Multicorrelator_8sc . . . 512
Fpga_spidev . . . 516
Fpga_Switch . . . 517
ftp_t . . . 519
Galileo_Almanac_Helper . . . 523
Galileo_Cnav_Message . . . 528
Galileo_Ephemeris_Store . . . 538
Galileo_Fnav_Message . . . 540
Galileo_HAS_data . . . 542
Galileo_HAS_page . . . 553
Galileo_Inav_Message . . . 555
Galileo_Iono . . . 558
Galileo_ISM . . . 560
Galileo_Reduced_CED . . . 564
Galileo_Utc_Model . . . 569
GalileoTowMapEntry . . . 574
Geohash . . . 576
GeoJSON_Printer . . . 577
geph_t . . . 578
Glonass_Gnav_Almanac . . . 580
Glonass_Gnav_Ephemeris . . . 584
Glonass_Gnav_Navigation_Message . . . 593
Glonass_Gnav_Utc_Model . . . 598
Gnss_Almanac . . . 603
  Beidou_Dnav_Almanac . . . 378
  Galileo_Almanac . . . 520
  Gps_Almanac . . . 648
Gnss_circular_deque< T > . . . 606
Gnss_Crypto . . . 607
Gnss_Ephemeris . . . 610
  Beidou_Cnav1_Ephemeris . . . 369
  Beidou_Dnav_Ephemeris . . . 383

- Galileo_Ephemeris . . . 533
- Gps_CNAV_Ephemeris . . . 652
- Gps_Ephemeris . . . 662

Gnss_Satellite . . . 618
Gnss_Sdr_Supl_Client . . . 622
Gnss_Signal . . . 630
Gnss_Synchro . . . 631
Gnss_Synchro_Udp_Sink . . . 638
GNSSBlockInterface . . . 638

- AcquisitionInterface . . . 341
  - PcpsAcquisitionAdapter . . . 828
  - PcpsAcquisitionAdapterCustom . . . 831
  - PcpsAcquisitionAdapterFpga . . . 834
- ArraySignalConditioner . . . 356
- BeamformerFilter . . . 365
- ByteToShort . . . 402
- ChannelInterface . . . 410
  - Channel . . . 403
- CshortToGrComplex . . . 432
- DirectResamplerConditioner . . . 442
- FirFilter . . . 497
- FreqXlatingFirFilter . . . 518
- IbyteToCbyte . . . 707
- IbyteToComplex . . . 708
- IbyteToCshort . . . 710
- IshortToComplex . . . 721
- IshortToCshort . . . 722
- MmseResamplerConditioner . . . 746
- NotchFilter . . . 768
- NotchFilterLite . . . 770
- ObservablesInterface . . . 790
  - HybridObservables . . . 705
- Pass_Through . . . 805
- PulseBlankingFilter . . . 853
- PvtInterface . . . 866
  - Rtklib_Pvt . . . 906
- SignalConditioner . . . 963
- SignalGenerator . . . 965
- SignalSourceInterface . . . 969
  - GenSignalSource . . . 574
  - SignalSourceBase . . . 967
    - Ad936xCustomSignalSource . . . 345
    - Adrv9361z7035SignalSourceFPGA . . . 347
    - BladerfSignalSource . . . 400
    - CustomUDPSignalSource . . . 437
    - DMASignalSourceFPGA . . . 464
    - Evk1029SignalSource . . . 476
    - FifoSignalSource . . . 482
    - FileSourceBase . . . 490
      - FileSignalSource . . . 488
      - FileTimestampSignalSource . . . 495
      - FourBitCpxFileSignalSource . . . 504
      - NTLabFileSignalSource . . . 774
      - NsrFileSignalSource . . . 772
      - SpirFileSignalSource . . . 978
      - TwoBitCpxFileSignalSource . . . 1014
      - TwoBitPackedFileSignalSource . . . 1017

FlexibandSignalSource . . . 499
Fmcomms2SignalSource . . . 501
Fmcomms5SignalSourceFPGA . . . 503
GSS6450FileSignalSource . . . 693
GnMaxSignalSource . . . 601
IONGSMSSignalSource . . . 719
LabsatSignalSource . . . 731
LimesdrSignalSource . . . 736
MAX2771EVKITSignalSourceFPGA . . . 745
MultichannelFileSignalSource . . . 756
OsmosdrSignalSource . . . 792
PlutosdrSignalSource . . . 839
PocketSdrSignalSource . . . 841
RawArraySignalSource . . . 878
RtlTcpSignalSource . . . 927
UhdSignalSource . . . 1020
ZmqSignalSource . . . 1030
TelemetryDecoderInterface . . . 1001
TelemetryDecoderAdapter . . . 999
TrackingInterface . . . 1012
DllPllTrackingAdapter . . . 460
DllPllTrackingAdapterFpga . . . 462
GalileoE1TcpConnectorTracking . . . 572
GpsL1CaDllPllTrackingGPU . . . 681
GpsL1CaGaussianTracking . . . 683
GpsL1CaKfTracking . . . 686
GpsL1CaTcpConnectorTracking . . . 688
GNSSFlowgraph . . . 641
GnssSdrLogSinkGuard . . . 645
GnssTime . . . 645
Gps_Acq_Assist . . . 646
Gps_CNAV_Eop . . . 650
Qzss_CNAV_Eop . . . 868
Gps_CNAV_Navigation_Message . . . 659
Gps_Iono . . . 668
Beidou_Dnav_Iono . . . 388
Gps_CNAV_Iono . . . 658
Qzss_CNAV_Iono . . . 869
Qzss_Iono . . . 871
Gps_Navigation_Message . . . 676
Gps_Utc_Model . . . 679
Gps_CNAV_Utc_Model . . . 661
Qzss_CNAV_Utc_Model . . . 870
Qzss_Utc_Model . . . 873
GPU_Complex . . . 690
GPU_Complex_Short . . . 691
Gpx_Printer . . . 692
gtime_t . . . 696
half_cyc_tag . . . 696
HAS_bias_corrections . . . 697
HAS_clock_corrections . . . 698
HAS_obs_corrections . . . 699
HAS_orbit_corrections . . . 699
Has_Simple_Printer . . . 700
std::hash< SensorIdentifier::value_type > . . . 701
HistogramBitSynchronizer . . . 701
INIReader . . . 711

IONGSMSChunkData . . . 716
IONGSMSChunkUnpackingCtx< WT > . . . 717
kernel_info_t . . . 723
Kf_Conf . . . 725
Kml_Printer . . . 730
lex_t . . . 732
lexeph_t . . . 733
lexion_t . . . 734
lexmsg_t . . . 735
absl::LogSink
  GnssSdrLogSink . . . 644
MACK_header . . . 738
Mack_lookup . . . 738
MACK_message . . . 739
MACK_tag_and_info . . . 740
MACK_tag_info . . . 741
matlab_type_traits< T > . . . 742
matlab_type_traits< double > . . . 742
matlab_type_traits< float > . . . 742
matlab_type_traits< int32_t > . . . 743
matlab_type_traits< int64_t > . . . 743
matlab_type_traits< uint32_t > . . . 744
matlab_type_traits< uint64_t > . . . 744
matlab_type_traits< uint8_t > . . . 744
ModelFunction . . . 748
Monitor_Ephemeris_Udp_Sink . . . 748
Monitor_Pvt . . . 748
Monitor_Pvt_Udp_Sink . . . 753
msm_h_t . . . 753
mt1_header . . . 755
Nav_Message_Packet . . . 758
Nav_Message_Udp_Sink . . . 761
nav_t . . . 761
Nmea_Printer . . . 767
Ntrip_Rtcm_Client . . . 777
Ntrip_Rtcm_Client_Config . . . 778
Ntrip_Rtcm_Client_Status . . . 780
Ntrip_Rtcm_Snapshot . . . 781
Ntrip_Rtk_Signal . . . 783
ntrip_t . . . 784
Obs_Conf . . . 787
obs_t . . . 788
obsd_t . . . 789
opt_t . . . 791
Osnma_Crypto_Material_Manager . . . 793
OSNMA_data . . . 794
OSNMA_DSM_Reader . . . 795
Osnma_Helper . . . 796
Osnma_Merkle_Tree_Material . . . 796
OSNMA_msg . . . 797
OSNMA_NavData . . . 800
OSNMA_NavDataManager . . . 803
Osnma_Public_Key_Material . . . 804
pclk_t . . . 807
pcps_acquisition_fpga . . . 813
pcv_t . . . 837
pcvs_t . . . 838
peph_t . . . 838

pppcorr_t . . . 842
pps_tcp_rx . . . 843
PpsSamplestamp . . . 844
prcopt_t . . . 844
Protobuf_Cleanup_Manager . . . 851
Osnma_Merkle_Tree_Material::PublicKeyEntry . . . 851
Pvt_Conf . . . 854
Pvt_Kf . . . 863
Pvt_Solution . . . 863
    Rtklib_Solver . . . 914
raw_t . . . 874
ReedSolomon . . . 880
Rinex_Printer . . . 881
Rtcm . . . 884
Rtcm_Printer . . . 899
rtcm_t . . . 901
rtk_t . . . 904
Rtklib_Tls_Client . . . 921
rtksvr_t . . . 922
Rtl_Tcp_Dongle_Info . . . 925
SatelliteVisibility . . . 928
Sbas_Ephemeris . . . 930
Sbas_Raw_Message . . . 933
Sbas_Rtklib_Corrections . . . 934
SbasL1DumpSession . . . 935
sbs_t . . . 936
sbsfcorr_t . . . 936
sbsigp_t . . . 937
sbsigpband_t . . . 938
sbsion_t . . . 938
sbslcorr_t . . . 939
sbsmsg_t . . . 940
sbssat_t . . . 940
sbssatp_t . . . 941
sdr_gnss_packet_t . . . 942
Secure_String . . . 943
IONGSMSFileSource::SegmentDescriptor . . . 944
SensorDataAggregator . . . 945
SensorDataConfiguration . . . 946
SensorDataFile . . . 946
SensorDataFileConfiguration . . . 947
SensorDataSample< DataType > . . . 948
SensorDataSourceConfiguration . . . 949
SensorDataType . . . 950
SensorIdentifier . . . 950
seph_t . . . 951
Serdes_Galileo_Eph . . . 952
Serdes_Gnss_Synchro . . . 953
Serdes_Gps_Eph . . . 955
Serdes_Monitor_Pvt . . . 957
Serdes_Nav_Message . . . 958
serial_t . . . 959
Signal_Enabled_Flags . . . 960
snrmask_t . . . 971
sol_t . . . 971
solbuf_t . . . 972
solopt_t . . . 974
solstat_t . . . 976

solstatbuf_t . . . 977
ssat_t . . . 980
ssr_t . . . 983
sta_t . . . 985
stec_t . . . 986
stream_cfg . . . 987
stream_t . . . 988
StringConverter . . . 990
gr::sync_block
  Evk1029Source . . . 478
  FifoReader . . . 481
  Gnss_Sdr_Timestamp . . . 629
  Gnss_Sdr_Valve . . . 629
  Gr_Complex_Ip_Packet_Source . . . 693
  IONGSMSFileSource . . . 718
  SensorDataSource . . . 949
  beamformer . . . 365
  bladerf_source . . . 399
  byte_x2_to_complex_byte . . . 401
  complex_byte_to_float_x2 . . . 419
  complex_float_to_complex_byte . . . 419
  conjugate_cc . . . 426
  conjugate_ic . . . 426
  conjugate_sc . . . 427
  cshort_to_float_x2 . . . 431
  cshort_to_gr_complex . . . 431
  rtklib_pvt_gs . . . 911
  rtl_tcp_signal_source_c . . . 926
  short_x2_to_cshort . . . 960
gr::sync_decimator
  gnss_sdr_sample_counter . . . 621
  interleaved_byte_to_complex_byte . . . 714
  interleaved_byte_to_complex_short . . . 715
  interleaved_short_to_complex_short . . . 715
gr::sync_interpolator
  unpack_2bit_samples . . . 1021
  unpack_byte_2bit_cpx_samples . . . 1022
  unpack_byte_2bit_samples . . . 1023
  unpack_byte_4bit_samples . . . 1023
  unpack_gss6450_samples . . . 1024
  unpack_intspir_1bit_samples . . . 1024
  unpack_ntlab_2bit_samples . . . 1025
  unpack_short_byte_samples . . . 1025
Tag . . . 990
Tcp_Communication . . . 993
Tcp_Packet_Data . . . 993
tcp_t . . . 994
tcpcli_t . . . 995
TcpCmdInterface . . . 995
tcpsvr_t . . . 996
tec_t . . . 996
tle_t . . . 1001
tled_t . . . 1002
Tlm_Conf . . . 1004
Tlm_CRC_Stats . . . 1006
TOW_to_trk . . . 1006
Monitor_Pvt::TrackedSatelliteInfo . . . 1007
Tracking_2nd_DLL_filter . . . 1009

Tracking_2nd_PLL_filter 1010
Tracking_FLL_PLL_filter 1010
Tracking_loop_filter 1011
TrackingCmd 1012
trop_t 1014
UnscentedFilter 1026
url_t 1026
v27_decision_t 1027
v27_poly_t 1027
v27_t 1027
Viterbi_Decoder 1028
Viterbi_Decoder_Sbas 1030

Chapter 8

Class Index

8.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

sdr_gnss_packet_t::[struct].sats . . . 331
Acq_Conf . . . 331
Acq_Conf_Fpga . . . 336
acquisition_impl_interface
This abstract class represents an interface to an acquisition GNSS block . . . 339
AcquisitionInterface
This abstract class represents an interface to an acquisition GNSS block . . . 341
ad936x_iio_custom . . . 343
ad936x_iio_samples . . . 344
ad936x_iio_source
This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex 344
Ad936xCustomSignalSource
This class instantiates the Ad936xCustom gnuradio signal source. It has support also for a customized Ad936xCustom firmware and signal source to support PPS samplestamp reading . 345
Adrv9361z7035SignalSourceFPGA . . . 347
Agnss_Ref_Location
Interface of an Assisted GNSS REFERENCE LOCATION storage . . . 349
Agnss_Ref_Time
Interface of an Assisted GNSS REFERENCE TIME storage . . . 350
Allocator . . . 351
alm_t . . . 352
ambc_t . . . 354
An_Packet_Printer
Class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol . . . 354
an_packet_t . . . 355
ArraySignalConditioner
This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal . . . 356
Bayesian_estimator
Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance) . . . 358
Bds3_B1c_AlmanacMedium . . . 359
Bds3_B1c_AlmanacReduced . . . 360
Bds3_B1c_Bgto . . . 361
Bds3_B1c_Eop . . . 362

Bds3_B1c_PageCommon . . . 363
Bds3_B1c_PageData . . . 364
beamformer
This class implements a real-time software-defined spatial filter using the CTTC GNSS experimental antenna array input and a set of dynamically reloadable weights . . . 365
BeamformerFilter
Interface of an adapter of a digital beamformer block to a GNSSBlockInterface . . . 365
beidou_b1c_telemetry_decoder_gs . . . 367
beidou_b2a_telemetry_decoder_gs . . . 368
Beidou_Cnav1_Ephemeris . . . 369
Beidou_Cnav1_Iono . . . 373
Beidou_Cnav1_Navigation_Message . . . 374
Beidou_Cnav1_PageData_Message . . . 375
Beidou_Cnav1_Utc_Model . . . 375
Beidou_Cnav2_Navigation_Message . . . 377
Beidou_Dnav_Almanac
This class is a storage for the BeiDou D1/D2 almanac . . . 378
Beidou_Dnav_Differential_Correction . . . 380
Beidou_Dnav_Differential_Corrections . . . 381
Beidou_Dnav_Ephemeris
This is a storage class for the Beidou SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0) . . . 383
Beidou_Dnav_Iono
This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1 . . . 388
Beidou_Dnav_Iono_Grid_Point . . . 390
Beidou_Dnav_Navigation_Message
This class decodes BeiDou D1 and D2 NAV data messages . . . 391
beidou_dnav_telemetry_decoder_gs
This class implements a block that decodes the BeiDou DNAV data, broadcast by both the B1I and B3I signals (frequency_band = 1 or 3, respectively) . . . 393
Beidou_Dnav_Utc_Model
This class is a storage for the BeiDou DNAV UTC Model . . . 395
BeidouLdpcGraph . . . 398
bladerf_source
This class implements a GNU Radio source block that streams RX samples from a bladeRF device (single RX channel) using libbladeRF's synchronous streaming API, converting the device's native BLADERF_FORMAT_SC16_Q11 samples to gr_complex . . . 399
BladerfSignalSource
This class instantiates the bladeRF signal source, streaming RX samples directly through libbladeRF (no gr-osmosdr or other intermediate driver required) . . . 400
byte_x2_to_complex_byte
This class adapts two signed char streams into a std::complex<signed char> stream . . . 401
ByteToShort
Adapts an 8-bits sample stream (IF) to a short int stream (IF) . . . 402
Channel
This class represents a GNSS channel. It wraps an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction through a Finite State Machine . . . 403
Channel_Event . . . 407
channel_msg_receiver_cc
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks . . . 407
channel_status_msg_receiver
GNU Radio block that receives asynchronous channel messages from tlm blocks . . . 408
ChannelFsm
This class implements a State Machine for channel . . . 409
ChannelInterface
This abstract class represents an interface to a channel GNSS block . . . 410

cl_fft_plan . . . 411
clFFT_Complex . . . 413
clFFT_Dim3 . . . 414
clFFT_SplitComplex . . . 414
cnav_msg_decoder_t . . . 415
cnav_msg_t . . . 415
cnav_v27_part_t . . . 416
Command_Event . . . 418
complex_byte_to_float_x2
This class adapts a std::complex<signed char> stream into two 16-bits (short) streams . . . 419
complex_float_to_complex_byte
This class adapts a gr_complex stream into a std::complex<signed char> stream . . . 419
Concurrent_Map< Data >
This class implements a thread-safe std::map . . . 420
Concurrent_Queue< Data >
This class implements a thread-safe std::queue . . . 421
HistogramBitSynchronizer::Config
Configuration parameters for HistogramBitSynchronizer . . . 422
ConfigurationInterface
This abstract class represents an interface to configuration parameters . . . 425
conjugate_cc
This class adapts a std::complex<short> stream into two 32-bits (float) streams . . . 426
conjugate_ic
This class adapts a std::complex<short> stream into two 32-bits (float) streams . . . 426
conjugate_sc
This class adapts a std::complex<short> stream into two 32-bits (float) streams . . . 427
ControlThread
This class represents the main thread of the application, so the name is ControlThread. This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions . . . 427
Cpu_Multicorrelator
Class that implements carrier wipe-off and correlators . . . 429
Cpu_Multicorrelator_16sc
Class that implements carrier wipe-off and correlators . . . 430
Cpu_Multicorrelator_Real_Codes
Class that implements carrier wipe-off and correlators . . . 430
cshort_to_float_x2
This class adapts a std::complex<short> stream into two 32-bits (float) streams . . . 431
cshort_to_gr_complex
This class adapts a short (16-bits) interleaved sample stream into a std::complex<float> stream 431
CshortToGrComplex
Adapts an 16-bits complex sample stream to a float complex stream . . . 432
CubatureFilter . . . 433
cuda_multicorrelator
Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators . 434
CudaPcpsEngine
Batched PCPS grid evaluation on a CUDA device . . . 434
CustomUDPSignalSource
This class reads from UDP packets, which streams interleaved I/Q samples over a network . . 437
Deleter . . . 438
dgps_t . . . 439
direct_resampler_conditioner_cb
This class implements a direct resampler conditioner for std::complex<signed char> . . . 440
direct_resampler_conditioner_cc
This class implements a direct resampler conditioner for complex data . . . 441
direct_resampler_conditioner_cs
This class implements a direct resampler conditioner for std::complex<short> . . . 441

DirectResamplerConditioner
Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface . 442
Dll_Pll_Conf . . . 444
Dll_Pll_Conf_Fpga . . . 450
dll_pll_veml_tracking
This class implements a code DLL + carrier PLL tracking block . . . 457
dll_pll_veml_tracking_fpga
This class implements a code DLL + carrier PLL tracking block . . . 458
DllPllTrackingAdapter
Base class providing shared logic for DLL+PLL tracking loop adapters for GNSS signals . . . 460
DllPllTrackingAdapterFpga . . . 462
DMASignalSourceFPGA . . . 464
DSM_dsm_header . . . 466
DSM_KROOT_message . . . 466
DSM_nma_header . . . 468
DSM_PKR_message . . . 469
eph_t . . . 470
erp_t . . . 475
erpd_t . . . 475
Evk1029SignalSource
This class reads a SAPHYRION EVK1029 raw capture file . . . 476
Evk1029Source
Reads a continuous, header-less EVK1029 raw capture file and unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first) . . . 478
Exponential_Smoother
Class that implements a first-order exponential smoother . . . 478
exterr_t . . . 480
fcbd_t . . . 480
FifoReader . . . 481
FifoSignalSource
Class that reads a sample stream from a Unix FIFO . . . 482
file_t . . . 484
FileConfiguration
This class is an implementation of the interface ConfigurationInterface . . . 486
FileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 488
FileSourceBase
Base class to file-oriented SignalSourceBase GNSS blocks . . . 490
FileTimestampSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 495
FirFilter
This class adapts a GNU Radio gr_fir_filter designed with pm_remez . . . 497
FlexibandSignalSource
This class configures and reads samples from Teleorbit Flexiband front-end. This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR) . . . 499
Fmcomms2SignalSource . . . 501
Fmcomms5SignalSourceFPGA . . . 503
FourBitCpxFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 504
Fpga_Acquisition
Class that implements carrier wipe-off and correlators . . . 507
Fpga_buffer_monitor
Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers 509
Fpga_DMA
Class that controls DMA transfers to the FPGA . . . 510
Fpga_dynamic_bit_selection
Controls dynamic bit selection in the FPGA . . . 512

Fpga_Multicorrelator_8sc
Class that implements carrier wipe-off and correlators . . . 512
Fpga_spidev . . . 516
Fpga_Switch
Class that controls the switch in the FPGA, which connects the FPGA acquisition and multicorrelator modules to either the DMA or the Analog Front-End . . . 517
FreqXlatingFirFilter
This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez . . . 518
ftp_t . . . 519
Galileo_Almanac
This class is a storage for the Galileo SV ALMANAC data . . . 520
Galileo_Almanac_Helper
This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD . . . 523
Galileo_Cnav_Message
This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022 . . . 528
Galileo_E1_Tcp_Connector_Tracking_cc
This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals . . . 529
galileo_e5a_noncoherentIQ_acquisition_caf_cc
This class implements a Parallel Code Phase Search Acquisition . . . 530
galileo_e6_has_msg_receiver
GNU Radio block that receives asynchronous Galileo HAS message pages from the telemetry blocks, stores them in memory, and decodes HAS messages when enough data have been received. The decoded HAS message is sent to the PVT block . . . 532
Galileo_Ephemeris . . . 533
Galileo_Ephemeris_Store . . . 538
Galileo_Fnav_Message
This class handles the Galileo F/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf . . . 540
Galileo_HAS_data
This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022. See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf . . . 542
Galileo_HAS_page
This class is a storage for Galileo HAS message page, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022 . . . 553
Galileo_Inav_Message
This class handles the Galileo I/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf . . . 555
Galileo_Iono
This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6 . . . 558
Galileo_ISM
This class is a storage for the GALILEO Integrity Support Message as described in Galileo ICD paragraph 5.2 . . . 560
galileo_pcps_8ms_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes) . . . 561
Galileo_Reduced_CED
This class manages the Galileo Reduced Clock and Ephemeris Data . . . 564
galileo_telemetry_decoder_gs
This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD . . . 567

galileo_tow_map . . . 568
Galileo_Utc_Model
This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD https://www.gsc-europa.eu/sites/default/files/sites/all/↩ files/Galileo_OS_SIS_ICD_v2.0.pdf paragraph 5.1.7 . . . 569
GalileoE1TcpConnectorTracking
This class implements a code DLL + carrier PLL tracking loop . . . 572
GalileoTowMapEntry . . . 574
GenSignalSource
This class wraps blocks that generates synthesized GNSS signal and filters the signal . . . 574
Geohash
Class for geohash encoding / decoding See https://en.wikipedia.org/wiki/↩ Geohash . . . 576
GeoJSON_Printer
Prints PVT solutions in GeoJSON format file . . . 577
geph_t . . . 578
Glonass_Gnav_Almanac
This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1) . . . 580
Glonass_Gnav_Ephemeris
This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1) . . . 584
Glonass_Gnav_Navigation_Message
This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1) . . . 593
glonass_gnav_telemetry_decoder_gs
This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals (frequency_band = 1 or 2, respectively) . . . 597
Glonass_Gnav_Utc_Model
This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1) . . . 598
GnMaxSignalSource
This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler . . . 601
Gnss_Almanac
Base class for GNSS almanac storage . . . 603
Gnss_circular_deque< T > . . . 606
Gnss_Crypto
Class implementing cryptographic functions for Navigation Message Authentication . . . 607
Gnss_Ephemeris
Base class for GNSS ephemeris storage . . . 610
Gnss_Satellite
This class represents a GNSS satellite . . . 618
gnss_sdr_fpga_sample_counter . . . 621
gnss_sdr_sample_counter . . . 621
Gnss_Sdr_Supl_Client
Class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library. . . . 622
gnss_sdr_time_counter . . . 628
Gnss_Sdr_Timestamp . . . 629
Gnss_Sdr_Valve
Implementation of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it . . . 629
Gnss_Signal
This class represents a GNSS signal . . . 630
Gnss_Synchro
This is the class that contains the information that is shared by the processing blocks . . . 631

gnss_synchro_monitor
This class implements a monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP 637
Gnss_Synchro_Udp_Sink
This class sends serialized Gnss_Synchro objects over UDP to one or multiple endpoints . . . 638
GNSSBlockInterface
This abstract class represents an interface to GNSS blocks 638
GNSSFlowgraph
This class represents a GNSS flow graph 641
GnssSdrLogSink 644
GnssSdrLogSinkGuard 645
GnssTime 645
Gps_Acq_Assist
This class is a storage for the GPS GSM RRLL acquisition assistance data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5) 646
Gps_Almanac
This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M 648
Gps_CNAV_Eop
Storage for the Earth orientation parameters broadcast in GPS/QZSS CNAV Message Type 32, as described in IS-GPS-200M 650
Gps_CNAV_Ephemeris
This is a storage class for the GPS CNAV ephemeris data as described in IS-GPS-200M . . . 652
Gps_CNAV_Iono
This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M 658
Gps_CNAV_Navigation_Message
This class decodes a GPS CNAV Data message as described in IS-GPS-200M 659
Gps_CNAV_Utc_Model
This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M 661
Gps_Ephemeris
This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200M 662
Gps_Iono
This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M 668
Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc
This class implements a DLL + PLL tracking loop block 670
Gps_L1_Ca_Gaussian_Tracking_cc
This class implements a DLL + PLL tracking loop block 671
Gps_L1_Ca_Tcp_Connector_Tracking_cc
This class implements a DLL + PLL tracking loop block 672
gps_l1_ca_telemetry_decoder_gs
This class implements a block that decodes the NAV data defined in IS-GPS-200M 672
gps_l2c_telemetry_decoder_gs
This class implements a block that decodes CNAV data defined in IS-GPS-200M 674
gps_l5_telemetry_decoder_gs
This class implements a GPS L5 Telemetry decoder 675
Gps_Navigation_Message
This class decodes a GPS NAV Data message as described in IS-GPS-200M 676
Gps_Utc_Model
This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M 679
GpsL1CaDllPllTrackingGPU
This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions 681
GpsL1CaGaussianTracking
This class implements a code DLL + carrier PLL tracking loop 683
GpsL1CaKfTracking
This class implements a code + carrier Kalman Filter tracking loop with VTL capabilities 686

GpsL1CaTcpConnectorTracking
This class implements a code DLL + carrier PLL tracking loop . . . 688
GPU_Complex . . . 690
GPU_Complex_Short . . . 691
Gpx_Printer
Prints PVT information to GPX format file . . . 692
Gr_Complex_Ip_Packet_Source . . . 693
GSS6450FileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 693
gtime_t . . . 696
half_cyc_tag . . . 696
HAS_bias_corrections . . . 697
HAS_clock_corrections . . . 698
HAS_obs_corrections . . . 699
HAS_orbit_corrections . . . 699
Has_Simple_Printer
Prints HAS messages content in a txt file. See HAS-SIS-ICD for a message description . . . 700
std::hash< SensorIdentifier::value_type > . . . 701
HistogramBitSynchronizer
Histogram-based navigation data bit synchronizer . . . 701
hybrid_observables_gs
This class implements a block that computes observables . . . 704
HybridObservables
This class implements an ObservablesInterface for observables of all kind of GNSS signals . . . 705
IbyteToCbyte . . . 707
IbyteToComplex
Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream . . . 708
IbyteToCshort
Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream 710
INIReader
Read an INI file into easy-to-access name/value pairs. (Note that I've gone for simplicity here rather than speed, but it should be pretty decent.) . . . 711
InMemoryConfiguration
This class is an implementation of the interface ConfigurationInterface . . . 712
interleaved_byte_to_complex_byte
This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::↩ complex<unsigned char>) . . . 714
interleaved_byte_to_complex_short
This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream 715
interleaved_short_to_complex_short
This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream 715
IONGSMSChunkData . . . 716
IONGSMSChunkUnpackingCtx< WT > . . . 717
IONGSMSFileSource . . . 718
IONGSMSSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 719
IshortToComplex
Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream . . . 721
IshortToCshort
Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream 722
kernel_info_t . . . 723
Kf_Conf . . . 725
kf_tracking
This class implements a code DLL + carrier PLL tracking block . . . 729
Kml_Printer
Prints PVT information to OGC KML format file (can be viewed with Google Earth) . . . 730
labsat23_source
This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex 730

LabsatSignalSource
This class reads samples stored in LabSat version 2, 3, and 3 Wideband format 731
lex_t 732
lexeph_t 733
lexion_t 734
lexmsg_t 735
LimesdrSignalSource
This class instantiates the LimeSDR gnuradio signal source. It has support also for a customized LimeSDR firmware and signal source to support PPS samplestamp reading 736
MACK_header 738
Mack_lookup 738
MACK_message 739
MACK_tag_and_info 740
MACK_tag_info 741
matlab_type_traits< T > 742
matlab_type_traits< double > 742
matlab_type_traits< float > 742
matlab_type_traits< int32_t > 743
matlab_type_traits< int64_t > 743
matlab_type_traits< uint32_t > 744
matlab_type_traits< uint64_t > 744
matlab_type_traits< uint8_t > 744
MAX2771EVKITSignalSourceFPGA 745
MmseResamplerConditioner
Interface of a MMSE resampler block adapter to a SignalConditionerInterface 746
ModelFunction 748
Monitor_Ephemeris_Udp_Sink 748
Monitor_Pvt
This class contains parameters and outputs of the PVT block 748
Monitor_Pvt_Udp_Sink 753
msm_h_t 753
mt1_header 755
MultichannelFileSignalSource
Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface 756
nav_message_monitor
GNU Radio block that receives asynchronous Nav_Message_Packet obkects from the telemetry blocks and sends them via UDP 758
Nav_Message_Packet 758
Nav_Message_Udp_Sink 761
nav_t 761
Nmea_Printer
This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA) . 767
Notch
This class implements a real-time software-defined multi state notch filter 768
NotchFilter 768
NotchFilterLite 770
NotchLite
This class implements a real-time software-defined multi state notch filter light version 771
NsrFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface 772
NTLabFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface 774
Ntrip_Rtcm_Client
Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster, preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced v1, optionally over TLS 777
Ntrip_Rtcm_Client_Config 778

Ntrip_Rtcm_Client_Status . . . 780
Ntrip_Rtcm_Snapshot . . . 781
Ntrip_Rtk_Signal . . . 783
ntrip_t . . . 784
Obs_Conf . . . 787
obs_t . . . 788
obsd_t . . . 789
ObservablesInterface
This abstract class represents an interface to an observables block . . . 790
opt_t . . . 791
OsmosdrSignalSource
This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki) . . . 792
Osnma_Crypto_Material_Manager . . . 793
OSNMA_data
This class handles ONSMA data See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf . . . 794
OSNMA_DSM_Reader . . . 795
Osnma_Helper . . . 796
Osnma_Merkle_Tree_Material . . . 796
OSNMA_msg
This class fills the OSNMA_msg structure with the data received from the telemetry blocks . . . 797
osnma_msg_receiver
GNU Radio block that receives asynchronous OSNMA messages from the telemetry blocks, stores them in memory, and decodes OSNMA info when enough data have been received. The decoded OSNMA data is sent to the PVT block . . . 798
OSNMA_NavData . . . 800
OSNMA_NavDataManager
Class for managing OSNMA navigation data . . . 803
Osnma_Public_Key_Material . . . 804
Pass_Through
This class implements a block that connects input and output (does nothing) . . . 805
pclk_t . . . 807
pcps_acquisition
This class implements a Parallel Code Phase Search Acquisition . . . 808
pcps_acquisition_fine_doppler_cc
This class implements a Parallel Code Phase Search Acquisition . . . 811
pcps_acquisition_fpga
This class implements a Parallel Code Phase Search Acquisition that uses the FPGA . . . 813
pcps_assisted_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition . . . 816
pcps_cccwsr_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme . . . 818
pcps_opencl_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition . . . 821
pcps_quicksync_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm . . . 823
pcps_tong_acquisition_cc
This class implements a Parallel Code Phase Search Acquisition with Tong algorithm . . . 826
PcpsAcquisitionAdapter
This class adapts a PCPS acquisition block to an AcquisitionInterface . . . 828
PcpsAcquisitionAdapterCustom
This class adapts a PCPS acquisition block to an AcquisitionInterface . . . 831
PcpsAcquisitionAdapterFpga . . . 834

pcv_t . . . 837
pcvs_t . . . 838
peph_t . . . 838
PlutosdrSignalSource . . . 839
PocketSdrSignalSource
This class instantiates the gr-pocketsdr signal source for the Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends . . . 841
pppcorr_t . . . 842
pps_tcp_rx . . . 843
PpsSamplestamp . . . 844
prcopt_t . . . 844
Protobuf_Cleanup_Manager
This class implements the Protocol Buffers library shared memory cleanup manager . . . 851
Osnma_Merkle_Tree_Material::PublicKeyEntry . . . 851
pulse_blanking_cc . . . 852
PulseBlankingFilter . . . 853
Pvt_Conf . . . 854
Pvt_Kf
Kalman Filter for Position and Velocity . . . 863
Pvt_Solution
Base class for a PVT solution . . . 863
PvtInterface
This class represents an interface to a PVT block . . . 866
Qzss_CNAV_Eop . . . 868
Qzss_CNAV_Iono
This class is a storage for the QZSS CNAV Klobuchar ionospheric model data as described in IS-QZSS-PNT-006 . . . 869
Qzss_CNAV_Utc_Model
This class is a storage for the QZSS CNAV UTC model data as described in IS-QZSS-PNT-006 870
Qzss_Iono
This class is a storage for the QZSS Klobuchar ionospheric model data as described in IS-↩ QZSS-PNT-006 . . . 871
Qzss_Utc_Model
This class is a storage for the QZSS UTC model data as described in IS-QZSS-PNT-006 . . . 873
raw_t . . . 874
RawArraySignalSource
This class reads samples from an antenna array RF front-end signal sampler . . . 878
ReedSolomon
Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where k=255-nroots is the information vector length and d=nroots+1 is the minimum Hamming distance, with symbols of 8 bits. It allows shortened RS codes . . . 880
Rinex_Printer
Class that handles the generation of Receiver INdependent EXchange format (RINEX) files . . 881
Rtcm
This class implements the generation and reading of some Message Types defined in the RTCM 3.2 Standard, plus some utilities to handle messages . . . 884
Rtcm_Printer
This class provides a implementation of a subset of the RTCM Standard 10403.2 messages . 899
rtcm_t . . . 901
rtk_t . . . 904
Rtklib_Pvt
This class implements a PvtInterface for the RTKLIB PVT block . . . 906
rtklib_pvt_gs
This class implements a block that computes the PVT solution using the RTKLIB integrated library . . . 911
Rtklib_Solver
This class implements a PVT solution based on RTKLIB . . . 914
Rtklib_Tls_Client . . . 921

rtksvr_t . . . 922
Rtl_Tcp_Dongle_Info
This class represents the dongle information which is sent by rtl_tcp . . . 925
rtl_tcp_signal_source_c
This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types . . . 926
RtlTcpSignalSource
This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see https://osmocom.org/projects/rtl-sdr/wiki) . . . 927
SatelliteVisibility
Opt-in acquisition search classification: visible, excluded, or unknown. Healthy satellites above the search mask are preferred; unknowns are searched less often. Banned PRNs are omitted, and tracking can override exclusion. Tick() refreshes on reference, fix, navigation-data, position, interval, or expiry changes. Expired data returns satellites to the unknown pool. Disabled by default (GNSS-SDR.enable_visibility_aware_search) . . . 928
Sbas_Ephemeris
This class stores SBAS SV ephemeris data . . . 930
sbas_l1_telemetry_decoder_gs
This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229 . . . 932
Sbas_Raw_Message
A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp . . . 933
Sbas_Rtklib_Corrections
Maintains RTKLIB SBAS state from timestamped raw L1 messages . . . 934
SbasL1DumpSession . . . 935
sbs_t . . . 936
sbsfcorr_t . . . 936
sbsigp_t . . . 937
sbsigpband_t . . . 938
sbsion_t . . . 938
sbslcorr_t . . . 939
sbsmsg_t . . . 940
sbssat_t . . . 940
sbssatp_t . . . 941
sdr_gnss_packet_t . . . 942
Secure_String
Value type for credential-bearing configuration fields . . . 943
IONGSMSFileSource::SegmentDescriptor . . . 944
SensorDataAggregator . . . 945
SensorDataConfiguration . . . 946
SensorDataFile . . . 946
SensorDataFileConfiguration . . . 947
SensorDataSample< DataType > . . . 948
SensorDataSource . . . 949
SensorDataSourceConfiguration . . . 949
SensorDataType . . . 950
SensorIdentifier . . . 950
seph_t . . . 951
Serdes_Galileo_Eph
This class implements serialization and deserialization of Galileo_Ephemeris using Protocol Buffers . . . 952
Serdes_Gnss_Synchro
This class implements serialization and deserialization of Gnss_Synchro objects using Protocol Buffers . . . 953
Serdes_Gps_Eph
This class implements serialization and deserialization of Gps_Ephemeris objects using Protocol Buffers . . . 955

Serdes_Monitor_Pvt
This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers . . . 957
Serdes_Nav_Message
This class implements serialization and deserialization of Nav_Message_Packet objects using Protocol Buffers . . . 958
serial_t . . . 959
short_x2_to_cshort
This class adapts two short streams into a std::complex<short> stream . . . 960
Signal_Enabled_Flags . . . 960
signal_generator_c
This class generates synthesized GNSS signal . . . 962
SignalConditioner
This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal . . . 963
SignalGenerator
This class generates synthesized GNSS signal . . . 965
SignalSourceBase . . . 967
SignalSourceInterface
This abstract class represents an interface to signal_source GNSS block . . . 969
snrmask_t . . . 971
sol_t . . . 971
solbuf_t . . . 972
solopt_t . . . 974
solstat_t . . . 976
solstatbuf_t . . . 977
SpirFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 978
ssat_t . . . 980
ssr_t . . . 983
sta_t . . . 985
stec_t . . . 986
stream_cfg . . . 987
stream_t . . . 988
StringConverter
Class that interprets the contents of a string and converts it into different types . . . 990
Tag . . . 990
Tcp_Communication
TCP communication class . . . 993
Tcp_Packet_Data
Class that implements a TCP data packet . . . 993
tcp_t . . . 994
tcpcli_t . . . 995
TcpCmdInterface . . . 995
tcpsvr_t . . . 996
tec_t . . . 996
telemetry_impl_interface
Common base class for telemetry decoder GNU Radio implementations . . . 997
TelemetryDecoderAdapter
Base class for Telemetry Decoder adapters . . . 999
TelemetryDecoderInterface
This abstract class represents an interface to a navigation GNSS block . . . 1001
tle_t . . . 1001
tled_t . . . 1002
Tlm_Conf . . . 1004
Tlm_CRC_Stats
Class that computes the telemetry CRC statistics . . . 1006
TOW_to_trk . . . 1006

Monitor_Pvt::TrackedSatelliteInfo
One tracked satellite/signal with its azimuth/elevation, whether it was combined with another signal of the same satellite (e.g. Galileo E1+E5a iono-free combination) and whether it was used in this fix. Signals are listed individually (one entry per satellite per signal): a combined satellite appears as two entries, both with combined = true . . . 1007
Tracking_2nd_DLL_filter
This class implements a 2nd order DLL filter for code tracking loop . . . 1009
Tracking_2nd_PLL_filter
This class implements a 2nd order PLL filter for carrier tracking loop . . . 1010
Tracking_FLL_PLL_filter
This class implements a hybrid FLL and PLL filter for tracking carrier loop . . . 1010
Tracking_loop_filter
This class implements a generic 1st, 2nd or 3rd order loop filter . . . 1011
TrackingCmd . . . 1012
TrackingInterface
This abstract class represents an interface to a tracking block . . . 1012
trop_t . . . 1014
TwoBitCpxFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 1014
TwoBitPackedFileSignalSource
Class that reads signals samples from a file and adapts it to a SignalSourceInterface . . . 1017
UhdSignalSource
This class reads samples from a UHD device (see http://code.ettus.com/↩ redmine/ettus/projects/uhd/wiki) . . . 1020
unpack_2bit_samples
This class takes 2 bit samples that have been packed into bytes or shorts as input and generates a byte for each sample. It generates eight times as much data as is input (every two bits become 16 bits) . . . 1021
unpack_byte_2bit_cpx_samples
This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples . . . 1022
unpack_byte_2bit_samples
This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples . . . 1023
unpack_byte_4bit_samples
This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples . . . 1023
unpack_gss6450_samples . . . 1024
unpack_intspir_1bit_samples
This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples . . . 1024
unpack_ntlab_2bit_samples
This class implements conversion between byte packet multichannel samples to 2bit samples 1 byte = 4 2bit samples . . . 1025
unpack_short_byte_samples
This class implements conversion between short packet samples to byte samples 1 short = 2 byte samples . . . 1025
UnscentedFilter . . . 1026
url_t . . . 1026
v27_decision_t . . . 1027
v27_poly_t . . . 1027
v27_t . . . 1027
Viterbi_Decoder
Class that implements a Viterbi decoder . . . 1028
Viterbi_Decoder_Sbas
Class that implements a Viterbi decoder . . . 1030
ZmqSignalSource
This class supports the following properties: . . . 1030

Chapter 9

File Index

9.1 File List

Here is a list of all documented files with brief descriptions:

pcps_acquisition_adapter.h
: Adapts a PCPS acquisition block to an AcquisitionInterface . . . 1033

pcps_acquisition_adapter_custom.h . . . 1035

pcps_acquisition_adapter_fpga.h
: Adapts an FPGA-offloaded PCPS acquisition block to an AcquisitionInterface . . . 1037

acquisition_impl_interface.h
: Header file of the interface to an acquisition implementation GNSS block . . . 1038

galileo_e5a_noncoherent_iq_acquisition_caf_cc.h
: Adapts a PCPS acquisition block to an AcquisitionInterface for Galileo E5a data and pilot Signals 1040

galileo_pcps_8ms_acquisition_cc.h
: This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes) . . . 1043

pcps_acquisition.h
: This class implements a Parallel Code Phase Search Acquisition . . . 1046

pcps_acquisition_fine_doppler_cc.h
: This class implements a Parallel Code Phase Search Acquisition with multi-dwells and fine Doppler estimation for GPS L1 C/A signal . . . 1051

pcps_acquisition_fpga.h
: This class implements a Parallel Code Phase Search Acquisition for the FPGA . . . 1054

pcps_assisted_acquisition_cc.h
: This class implements a Parallel Code Phase Search Acquisition with assistance and multi-dwells . . . 1057

pcps_cccwsr_acquisition_cc.h
: This class implements a Parallel Code Phase Search acquisition with Coherent Channel Combining With Sign Recovery scheme . . . 1060

pcps_opencl_acquisition_cc.h
: This class implements a Parallel Code Phase Search Acquisition using OpenCL to offload some functions to the GPU . . . 1063

pcps_quicksync_acquisition_cc.h
: This class implements a Parallel Code Phase Search Acquisition with the QuickSync Algorithm 1067

pcps_tong_acquisition_cc.h
: This class implements a Parallel Code Phase Search Acquisition with Tong algorithm . . . 1071

acq_conf.h
: Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm . . . 1074

acq_conf_fpga.h
Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm running in the FPGA . . . 1076
cuda_pcps_engine.h
GPU (CUDA + cuFFT) implementation of the Parallel Code Phase Search grid . . . 1078
fpga_acquisition.h
Highly optimized FPGA vector correlator class . . . 1080
channel.h
Interface of a GNSS channel . . . 1083
channel_fsm.h
Interface of the State Machine for channel . . . 1085
channel_msg_receiver_cc.h
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks . . . 1087
array_signal_conditioner.h
It wraps blocks to change data type, filter and resample input data, adapted to array receiver . 1088
signal_conditioner.h
It wraps blocks to change data type, filter and resample input data . . . 1090
byte_to_short.h
Adapts an 8-bits sample stream (IF) to a short int stream (IF) . . . 1091
cshort_to_grcomplex.h
Adapts an 16-bits complex sample stream to a float complex stream . . . 1093
ibyte_to_cbyte.h
Adapts an I/Q interleaved byte (unsigned char) sample stream into a std::complex<unsigned char> stream . . . 1094
ibyte_to_complex.h
Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream . . . 1096
ibyte_to_cshort.h
Adapts a short interleaved sample stream into a std::complex<short> stream . . . 1097
ishort_to_complex.h
Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream . . . 1099
ishort_to_cshort.h
Adapts a short interleaved sample stream into a std::complex<short> stream . . . 1100
cshort_to_gr_complex.h
Adapts a complex short (16 + 16 bits) sample stream into a std::complex<float> stream (32 + 32 bits) . . . 1102
interleaved_byte_to_complex_byte.h
Adapts an 8-bits interleaved sample stream into a 16-bits complex stream . . . 1103
interleaved_byte_to_complex_short.h
Adapts a byte (8-bits) interleaved sample stream into a std::complex<short> stream . . . 1104
interleaved_short_to_complex_short.h
Adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream . . . 1106
beamformer_filter.h
Interface of an adapter of a digital beamformer . . . 1107
fir_filter.h
Adapts a gnuradio gr_fir_filter designed with pm_remez . . . 1108
freq_xlating_fir_filter.h
Adapts a gnuradio gr_freq_xlating_fir_filter designed with gr_remez . . . 1111
notch_filter.h
Adapter of a multistate Notch filter . . . 1113
notch_filter_lite.h
Adapts a light version of a multistate notch filter . . . 1114
pulse_blanking_filter.h
Instantiates the GNSS-SDR pulse blanking filter . . . 1115
beamformer.h
Simple spatial filter using RAW array input and beamforming coefficients . . . 1117
notch_cc.h
Implements a notch filter algorithm . . . 1118

notch_lite_cc.h
Implements a notch filter light algorithm . . . 1120
pulse_blanking_cc.h
Implements a pulse blanking algorithm . . . 1122
beidou_b1c_signal_replica.h
Library for BeiDou B1C signal replica generation . . . 1123
beidou_b1i_signal_replica.h
This file implements various functions for BeiDou B1I signal replica generation . . . 1125
beidou_b2a_signal_replica.h
BeiDou B2a ranging-code replica (BDS-SIS-ICD-B2a-1.0 Tables 5-2, 5-3) . . . 1126
beidou_b3i_signal_replica.h
This file implements various functions for BeiDou B3I signal replica generation . . . 1127
byte_x2_to_complex_byte.h
Adapts two signed char streams into a std::complex<signed char> stream . . . 1128
complex_byte_to_float_x2.h
Adapts a std::complex<signed char> stream into two 16-bits (short) streams . . . 1130
complex_float_to_complex_byte.h
Adapts a gr_complex stream into a std::complex<signed char> stream . . . 1131
conjugate_cc.h
Conjugate a stream of gr_complex . . . 1132
conjugate_ic.h
Conjugate a stream of lv_8sc_t (std::complex<char>) . . . 1133
conjugate_sc.h
Conjugate a stream of lv_16sc_t (std::complex<short>) . . . 1135
cshort_to_float_x2.h
Adapts a std::complex<short> stream into two float streams . . . 1136
dump_logger_helper.h
Utility functions for logging to a file . . . 1137
galileo_e1_signal_replica.h
This library implements various functions for Galileo E1 signal replica generation . . . 1138
galileo_e5_signal_replica.h
This library implements various functions for Galileo E5 signal replica generation . . . 1140
galileo_e6_signal_replica.h
This library implements various functions for Galileo E6 signal replica generation . . . 1142
geofunctions.h
A set of coordinate transformations functions and helpers, some of them migrated from MATLAB, for geographic information systems . . . 1144
glonass_l1_signal_replica.h . . . 1147
glonass_l2_signal_replica.h
This file implements various functions for GLONASS L2 CA signal replica generation . . . 1148
gnss_circular_deque.h
This class implements a circular deque for Gnss_Synchro . . . 1149
gnss_sdr_create_directory.h
Create a directory . . . 1151
gnss_sdr_fft.h
Helper file for FFT interface . . . 1152
gnss_sdr_filesystem.h
Helper file for fylesystem library interface . . . 1154
gnss_sdr_flags.h
Helper file for gnss-sdr commandline flags . . . 1155
gnss_sdr_make_unique.h
This file implements std::make_unique for C++11 . . . 1159
gnss_sdr_string_literals.h
This file implements the ""s operator for std::string in C++11, and puts it into the std::string_↩ literals namespace. This is already implemented in C++14, so this is only compiled when using C++11. The .cc file is required for avoiding the duplication of symbols . . . 1161

gnss_signal_replica.h
This library gathers a few functions used for GNSS signal replica generation regardless of system used . . . 1161
gnss_time.h
Class that stores both the receiver time, relative to the receiver start and the GNSS time (absolute) . . . 1163
gps_l2c_signal_replica.h
This file implements signal generators for GPS L2C signals . . . 1164
gps_l5_signal_replica.h
This file implements signal generators for GPS L5 signals . . . 1165
gps_sdr_signal_replica.h
This file implements functions for GPS L1 C/A signal replica generation . . . 1167
item_type_helpers.h
Utility functions for converting between item types . . . 1168
matlab_writter_helper.h
Utility functions for logging to a matlab file . . . 1170
clFFT.h
FFT in OpenCL . . . 1172
fft_base_kernels.h
FFT base kernels for OpenCL . . . 1174
fft_internal.h
Internals of FFT for OpenCL . . . 1178
pass_through.h
Interface of a block that just puts its input in its output . . . 1180
qzss_signal_replica.h
This file implements signal generators for QZSS signals . . . 1181
rtklib.h
Main header file for the rtklib library . . . 1183
rtklib_conversions.h
GNSS-SDR to RTKLIB data structures conversion functions . . . 1209
rtklib_ephemeris.h
Satellite ephemeris and clock functions . . . 1212
rtklib_ionex.h
Ionex functions . . . 1214
rtklib_lambda.h
Integer ambiguity resolution . . . 1217
rtklib_pntpos.h
Standard code-based positioning . . . 1219
rtklib_ppp.h
Precise Point Positioning . . . 1223
rtklib_preceph.h
Precise ephemeris and clock functions . . . 1229
rtklib_rtcm.h
RTCM functions headers . . . 1231
rtklib_rtcm2.h
RTCM v2 functions headers . . . 1233
rtklib_rtcm3.h
RTCM v3 functions headers . . . 1234
rtklib_rtkcmn.h
Rtklib common functions . . . 1240
rtklib_rtkpos.h
Rtklib ppp-related functions . . . 1248
rtklib_rtksvr.h
Rtk server functions . . . 1252
rtklib_sbas.h
Sbas functions . . . 1255
rtklib_secure_wipe.h
Single declaration of the credential-memory wipe primitive . . . 1259

rtklib_solution.h
Solution functions headers . . . 1260
rtklib_stream.h
Streaming functions . . . 1264
rtklib_tides.h
Tidal displacement corrections . . . 1270
rtklib_tls.h
TLS 1.2-or-newer client transport for RTKLIB streams . . . 1272
sbas_signal_replica.h
This file implements signal generators for SBAS L1 signals . . . 1274
sensor_data_aggregator.h
Aggregates sensor samples from gnu radio stream tags into typed lists for easy access . . . 1275
sensor_data_file.h
Provides a simple abstraction for reading contiguous binary data from a file . . . 1276
sensor_data_resampler.h
Updates timestamp within sensor data tags. To be used in resampler blocks . . . 1278
sensor_data_source.h
GNURadio block that adds extra data to the sample stream . . . 1279
sensor_data_source_configuration.h . . . 1280
sensor_data_type.h . . . 1282
sensor_identifier.h . . . 1283
short_x2_to_cshort.h
Adapts two short streams into a std::complex<short> stream . . . 1284
trackingcmd.h
Class that stores information to update the GNSS signal tracking estimations . . . 1285
hybrid_observables.h
Implementation of an adapter of an observables block accepting all kind of signals to a ObservablesInterface . . . 1286
hybrid_observables_gs.h
Interface of the observables computation block . . . 1288
obs_conf.h
Class that contains all the configuration parameters for generic observables block . . . 1292
rtklib_pvt.h
Interface of a Position Velocity and Time computation block . . . 1293
rtklib_pvt_gs.h
Interface of a Position Velocity and Time computation block . . . 1297
an_packet_printer.h
Interface of a class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol . . . 1303
galileo_ephemeris_store.h
Source-aware storage for Galileo I/NAV and F/NAV ephemerides . . . 1305
geohash.h
Interface of a class that encodes / decodes geohashes . . . 1308
geojson_printer.h
Interface of a class that prints PVT solutions in GeoJSON format . . . 1309
gpx_printer.h
Interface of a class that prints PVT information to a gpx file . . . 1310
has_simple_printer.h
Interface of a class that prints HAS messages content in a txt file . . . 1311
kml_printer.h
Interface of a class that prints PVT information to a kml file . . . 1313
monitor_ephemeris_udp_sink.h
Interface of a class that sends serialized Gps_Ephemeris and Galileo_Ephemeris objects over udp to one or multiple endpoints . . . 1314
monitor_pvt.h
Interface of the Monitor_Pvt class . . . 1315

monitor_pvt_udp_sink.h
Interface of a class that sends serialized Monitor_Pvt objects over udp to one or multiple endpoints . . . 1318
nmea_printer.h
Interface of a NMEA 2.1 printer for GNSS-SDR This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA). See https://www.nmea.org/ for the NMEA 183 standard . . . 1320
ntrip_rtcm_client.h
NTRIP client for RTCM3 fixed-base corrections . . . 1321
ntrip_rtk_signals.h
Signal table of the NTRIP RTK path . . . 1325
pvt_conf.h
Class that contains all the configuration parameters for the PVT block . . . 1329
pvt_kf.h
Kalman Filter for Position and Velocity . . . 1331
pvt_solution.h
Interface of a base class for a PVT solution . . . 1332
rinex_printer.h
Interface of a RINEX 2.11 / 3.02 / 4.02 printer See https://files.igs.org/pub/data/format/rinex_4.02.pdf . . . 1334
rtcm.h
Interface for the RTCM 3.2 Standard . . . 1339
rtcm_printer.h
Interface of a RTCM 3.2 printer for GNSS-SDR This class provides a implementation of a subset of the RTCM Standard 10403.2 for Differential GNSS Services . . . 1361
rtklib_solver.h
PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures 1364
sbas_rtklib_corrections.h
Bridge between raw SBAS messages and RTKLIB correction state . . . 1370
secure_string.h
Self-scrubbing value type for credential-bearing configuration fields . . . 1371
serdes_galileo_eph.h
Serialization / Deserialization of Galileo_Ephemeris objects using Protocol Buffers . . . 1373
serdes_gps_eph.h
Serialization / Deserialization of Gps_Ephemeris objects using Protocol Buffers . . . 1376
serdes_monitor_pvt.h
Serialization / Deserialization of Monitor_Pvt objects using Protocol Buffers . . . 1378
signal_enabled_flags.h
Class to check the enabled signals . . . 1381
direct_resampler_conditioner.h
Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface . 1383
mmse_resampler_conditioner.h
Interface of an adapter of a mmse resampler conditioner block to a SignalConditionerInterface 1384
direct_resampler_conditioner_cb.h
Nearest neighborhood resampler with std::complex<signed char> input and std::complex<signed char> output . . . 1386
direct_resampler_conditioner_cc.h
Nearest neighborhood resampler with gr_complex input and gr_complex output . . . 1388
direct_resampler_conditioner_cs.h
Nearest neighborhood resampler with std::complex<short> input and std::complex<short> output . . . 1389
signal_generator.h
Adapter of a class that generates synthesized GNSS signal . . . 1391
signal_generator_c.h
GNU Radio source block that generates synthesized GNSS signal . . . 1392

ad936x_custom_signal_source.h
A direct IIO custom front-end gnss-sdr signal source for the AD936x AD front-end family with special FPGA custom functionalities . . . 1395
adrv9361_z7035_signal_source_fpga.h
Signal source for the Analog Devices ADRV9361-Z7035 evaluation board directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file . . . 1398
bladerf_signal_source.h
Signal source for the Nuand bladeRF family of front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), using libbladeRF directly . . . 1400
custom_udp_signal_source.h
Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap) . . . 1401
dma_signal_source_fpga.h
Signal source for a DMA connected directly to FPGA accelerators. This source implements only the DMA control. It is NOT compatible with conventional SDR acquisition and tracking blocks . 1403
evk1029_signal_source.h
SAPHYRION EVK1029 signal source. Reads a continuous, header-less raw capture file (OBA-encoded 4-bit samples) directly, without going through the generic XML-metadata-driven ION↩ _GSMS_Signal_Source path . . . 1405
fifo_signal_source.h
Header file of the class for retrieving samples through a Unix FIFO . . . 1407
file_signal_source.h
Interface of a class that reads signals samples from a file and adapts it to a SignalSourceInterface 1408
file_source_base.h
Header file of the base class to file-oriented signal_source GNSS blocks . . . 1410
file_timestamp_signal_source.h
This class reads samples stored in a file and generate stream tags with its timestamp information stored in separated file . . . 1413
flexiband_signal_source.h
Ignal Source adapter for the Teleorbit Flexiband front-end device. This adapter requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR) . . . 1414
fmcomms2_signal_source.h
Interface to use SDR hardware based in FMCOMMS2 driver from analog devices, for example FMCOMMS4 and ADALM-PLUTO (PlutoSdr) . . . 1416
fmcomms5_signal_source_fpga.h
Signal source for the Analog Devices FMCOMMS5 directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file . . . 1418
four_bit_cpx_file_signal_source.h
Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface . . . 1420
gen_signal_source.h
It wraps blocks that generates synthesized GNSS signal and filters it . . . 1421
gnmax_signal_source.h
GnMAX2769 USB dongle GPS RF front-end signal sampler driver . . . 1423
gss6450_file_signal_source.h
Implementation of a class that reads GSS6450 sample files and adapts it to a SignalSourceInterface 1424
ion_gsms_signal_source.h
GNSS-SDR Signal Source that reads sample streams following ION's GNSS-SDR metadata standard . . . 1427
labsat_signal_source.h
LabSat version 2, 3, and 3 Wideband format reader . . . 1429
limesdr_signal_source.h . . . 1431

max2771_evkit_signal_source_fpga.h
Signal source for the MAX2771EVKIT evaluation board connected directly to FPGA accelerators. This source implements only the MAX2771 control. It is NOT compatible with conventional SDR acquisition and tracking blocks . . . 1432
multichannel_file_signal_source.h
Implementation of a class that reads signals samples from files at different frequency band and adapts it to a SignalSourceInterface . . . 1435
nsr_file_signal_source.h
Implementation of a class that reads signals samples from a NSR 2 bits sampler front-end file and adapts it to a SignalSourceInterface. More information about the front-end here http://www.ifen.com/products/sx-scientific-gnss-solutions/↩ nsr-software-receiver.html . . . 1437
ntlab_file_signal_source.h
Interface of a class that reads signal samples from a file. Each sample is two bits from multiple channels . . . 1438
osmosdr_signal_source.h
Signal source wrapper for OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/↩ projects/rtl-sdr/wiki for more information) . . . 1440
plutosdr_signal_source.h
Signal source for PlutoSDR . . . 1441
pocket_sdr_signal_source.h
Signal source for Pocket SDR front-ends . . . 1443
raw_array_signal_source.h
CTTC Experimental GNSS 8 channels array signal source . . . 1445
rtl_tcp_signal_source.h
Signal source which reads from rtl_tcp. (see https://osmocom.org/projects/↩ rtl-sdr/wiki for more information) . . . 1446
signal_source_base.h
Header file of the base class to signal_source GNSS blocks . . . 1448
spir_file_signal_source.h
Implementation of a class that reads signals samples from a SPIR file and adapts it to a SignalSourceInterface . . . 1449
two_bit_cpx_file_signal_source.h
Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface . . . 1450
two_bit_packed_file_signal_source.h
Interface of a class that reads signals samples from a file. Each sample is two bits, which are packed into bytes or shorts . . . 1452
uhd_signal_source.h
Interface for the Universal Hardware Driver signal source . . . 1454
zmq_signal_source.h
Signal source which reads from ZeroMQ . . . 1455
ad936x_iio_source.h
A direct IIO custom front-end gnss-sdr signal gnuradio block for the AD936x AD front-end family with special FPGA custom functionalities . . . 1457
bladerf_source.h
GNU Radio source block for Nuand's bladeRF front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), implemented directly against libbladeRF . . . 1460
evk1029_source.h
Unpacks the SAPHYRION EVK1029 raw capture files (continuous stream of OBA-encoded 4-bit samples, 16 samples packed per little-endian 64-bit word, no periodic block header) . . . 1462
fifo_reader.h
Header file to retrieve samples from an existing Unix FIFO . . . 1464
gr_complex_ip_packet_source.h
Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap) . . . 1466

ion_gsms.h
GNU Radio block that reads a Block from a file following ION's GNSS-SDR metadata standard 1467
labsat23_source.h
Unpacks capture files in the LabSat 2 (ls2), LabSat 3 (ls3), LabSat 3 Wideband (LS3W), and Labsat 4 (ls4) formats . . . 1469
rtl_tcp_signal_source_c.h
Interface of an rtl_tcp signal source reader . . . 1472
unpack_2bit_samples.h
Unpacks 2 bit samples samples may be packed in any of the following ways: 1) Into bytes [item == byte] 1a) Big endian ordering within the byte 1b) Little endian ordering within the byte 2) Into shorts [item == short] 2a) Big endian ordering of bytes, big endian within the byte 2b) Big endian ordering of bytes, little endian within the byte 2c) Little endian ordering of bytes, big endian within the byte 2d) Little endian ordering of bytes, little endian within the byte . . . 1475
unpack_byte_2bit_cpx_samples.h
Unpacks byte samples to 2 bits complex samples. Packing Order Most Significant Nibble - Sample n Least Significant Nibble - Sample n+1 Packing order in Nibble Q1 Q0 I1 I0 . . . 1477
unpack_byte_2bit_samples.h
Unpacks byte samples to NSR 2 bits samples . . . 1478
unpack_byte_4bit_samples.h
Unpacks byte samples to 4 bits samples. Packing Order Packing order in Nibble I0 I1 I2 I3 I0 I1 I2 I3 . . . 1480
unpack_gss6450_samples.h
Unpacks GSS6450 int samples . . . 1481
unpack_intspir_1bit_samples.h
Unpacks SPIR int samples to NSR 1 bit samples . . . 1482
unpack_ntlab_2bit_samples.h
Unpacks multichannel 2-bit samples into 4 real-valued floats per input byte . . . 1484
unpack_short_byte_samples.h . . . 1485
ad9361_manager.h
An Analog Devices AD9361 front-end configuration library wrapper for configure some functions via iiod link . . . 1486
ad936x_iio_custom.h
A direct IIO custom front-end driver for the AD936x AD front-end family with special FPGA custom functionalities . . . 1489
ad936x_iio_samples.h
A class that holds a custom sample buffer for Analog Devices AD936x family front-ends . . . 1491
fpga_buffer_monitor.h
Check receiver buffer overflow and monitor the status of the receiver buffers . . . 1492
fpga_dma-proxy.h
FPGA DMA control. This code is based in the Xilinx DMA proxy test application: https://github.com/Xilinx-Wiki-Projects/software-prototypes/tree/master/linux-user-space-dma/Software . . . 1494
fpga_dynamic_bit_selection.h
Dynamic bit selection in the received signal . . . 1496
fpga_spidev.h
FPGA SPI control . . . 1498
fpga_switch.h
Switch that connects the HW accelerator queues to the analog front end or the DMA . . . 1499
gnss_sdr_timestamp.h
GNURadio block that adds to sample stream timestamp metadata information stored on a separated file . . . 1501
gnss_sdr_valve.h
Interface of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it . . . 1502
ion_gsms_chunk_data.h
Holds logic for reading and decoding samples from a chunk . . . 1504
ion_gsms_chunk_unpacking_ctx.h
Holds state and provides utilities for unpacking samples from a chunk . . . 1508

ion_gsms_stream_encodings.h
Implements look up tables for all encodings in the standard . . . 1509
pps_samplestamp.h
A simple container for the sample count associated to PPS rising edge . . . 1515
ppstcprx.h
TCP client class for front-end PPS samplestamp information reception . . . 1516
rtl_tcp_commands.h
Defines structures and constants for communicating with rtl_tcp . . . 1517
rtl_tcp_dongle_info.h
Interface for a structure sent by rtl_tcp defining the hardware . . . 1518
telemetry_decoder_adapter.h
Common functionality for telemetry decoder adapters . . . 1520
beidou_b1c_telemetry_decoder_gs.h
BeiDou B1C B-CNAV1 telemetry decoder block . . . 1521
beidou_b2a_telemetry_decoder_gs.h
BeiDou B2a B-CNAV2 telemetry decoder . . . 1523
beidou_dnav_telemetry_decoder_gs.h
Implementation of a BeiDou DNAV data decoder block for the B1I and B3I signals . . . 1525
galileo_telemetry_decoder_gs.h
Implementation of a Galileo unified INAV and FNAV message demodulator block . . . 1527
glonass_gnav_telemetry_decoder_gs.h
Implementation of a GLONASS GNAV data decoder block for the L1 and L2 C/A signals . . . 1530
gps_l1_ca_telemetry_decoder_gs.h
Interface of a NAV message demodulator block based on Kay Borre book MATLAB-based GPS receiver . . . 1532
gps_l2c_telemetry_decoder_gs.h
Interface of a CNAV message demodulator block . . . 1534
gps_l5_telemetry_decoder_gs.h
Interface of a CNAV message demodulator block . . . 1536
sbas_l1_telemetry_decoder_gs.h
Interface of a SBAS telemetry data decoder block . . . 1538
telemetry_impl_interface.h
Base class for telemetry decoder GNU Radio blocks . . . 1541
galileo_tow_utils.h
Small helpers for Galileo TOW arithmetic in telemetry decoder blocks . . . 1543
bits.h
Utilities for bit manipulation of the libswiftnav library . . . 1545
cnav_msg.h
Utilities for CNAV message manipulation of the libswiftnav library . . . 1547
edc.h
Utilities for CRC computation of the libswiftnav library . . . 1549
fec.h
Utilities for the convolutional encoder of the libswiftnav library . . . 1550
swift_common.h
Common definitions used throughout the libswiftnav library . . . 1552
tlm_conf.h
Class that contains all the configuration parameters for generic telemetry decoder block . . . 1553
tlm_crc_stats.h
Class that computes the telemetry CRC statistics . . . 1554
tlm_utils.h
Utilities for the telemetry decoder blocks . . . 1555
tow_utils.h
Small helpers for TOW arithmetic shared by telemetry decoder blocks . . . 1556
viterbi_decoder.h
Class that implements a Viterbi decoder . . . 1559
viterbi_decoder_sbas.h
Interface of a Viterbi decoder class based on the Iterative Solutions Coded Modulation Library by Matthew C. Valenti . . . 1561

dll_pll_tracking_adapter.h
Base class providing shared logic for DLL+PLL VEML tracking adapters . . . 1563
dll_pll_tracking_adapter_fpga.h
Adapts an FPGA-offloaded DLL/PLL tracking block to a TrackingInterface . . . 1565
galileo_e1_tcp_connector_tracking.h
Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for Galileo E1 to a TrackingInterface . . . 1566
gps_l1_ca_dll_pll_tracking_gpu.h
Implementation of an adapter of a DLL+PLL tracking loop block using GPU accelerated functions for GPS L1 C/A to a TrackingInterface . . . 1568
gps_l1_ca_gaussian_tracking.h
Interface of an adapter of a DLL + Kalman carrier tracking loop block for GPS L1 C/A signals . 1570
gps_l1_ca_kf_tracking.h
Interface of an adapter of a code + carrier Kalman Filter tracking loop with VTL capabilities block for GPS L1 C/A to a TrackingInterface . . . 1571
gps_l1_ca_tcp_connector_tracking.h
Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for GPS L1 C/A to a TrackingInterface . . . 1573
dll_pll_veml_tracking.h
Implementation of a code DLL + carrier PLL tracking block . . . 1575
dll_pll_veml_tracking_fpga.h
Implementation of a code DLL + carrier PLL tracking block using an FPGA . . . 1579
galileo_e1_tcp_connector_tracking_cc.h
Interface of a TCP connector block based on code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals . . . 1583
gps_l1_ca_dll_pll_tracking_gpu_cc.h
Implementation of a code DLL + carrier PLL tracking block, GPU ACCELERATED . . . 1586
gps_l1_ca_gaussian_tracking_cc.h
Interface of a processing block of a DLL + Kalman carrier tracking loop for GPS L1 C/A signals 1589
gps_l1_ca_tcp_connector_tracking_cc.h
Interface of a TCP connector block based on code DLL + carrier PLL . . . 1592
kf_tracking.h . . . 1595
bayesian_estimation.h
Interface of a library with Bayesian noise statistic estimation . . . 1597
bit_synchronizer.h
Histogram-based bit-edge synchronizer for GNSS prompt correlator outputs . . . 1599
cpu_multicorrelator.h
High optimized CPU vector multiTAP correlator class . . . 1604
cpu_multicorrelator_16sc.h
Highly optimized CPU vector multiTAP correlator class for lv_16sc_t (short int complex) . . . 1605
cpu_multicorrelator_real_codes.h
Highly optimized CPU vector multiTAP correlator class using real-valued local codes . . . 1606
cuda_multicorrelator.h
Highly optimized CUDA GPU vector multiTAP correlator class . . . 1608
dll_pll_conf.h
Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL . . . 1611
dll_pll_conf_fpga.h
Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL for the FPGA . . . 1612
exponential_smoother.h
Class that implements an exponential smoother . . . 1614
fpga_multicorrelator.h
FPGA vector correlator class . . . 1615
kf_conf.h . . . 1619
lock_detectors.h
Interface of a library with a set of code and carrier phase lock detectors . . . 1620

nonlinear_tracking.h
Interface of a library for nonlinear tracking algorithms . . . 1622
tcp_communication.h
Interface of the TCP communication class . . . 1624
tcp_packet_data.h
Interface of the TCP data packet class . . . 1626
tracking_2nd_DLL_filter.h
Interface of a 2nd order DLL filter for code tracking loop . . . 1627
tracking_2nd_PLL_filter.h
Interface of a 2nd order PLL filter for carrier tracking loop . . . 1628
tracking_discriminators.h
Interface of a library with a set of code tracking and carrier tracking discriminators . . . 1629
tracking_FLL_PLL_filter.h
Interface of a hybrid FLL and PLL filter for tracking carrier loop . . . 1633
tracking_loop_filter.h
Generic 1st to 3rd order loop filter implementation . . . 1634
acquisition_interface.h
Header file of the interface to an acquisition GNSS block . . . 1635
channel_interface.h
This class represents an interface to a channel GNSS block . . . 1637
configuration_interface.h
This class represents an interface to configuration parameters . . . 1638
gnss_block_interface.h
This interface represents a GNSS block . . . 1639
observables_interface.h
This class represents an interface to an Observables block . . . 1641
pvt_interface.h
This class represents an interface to a PVT block . . . 1642
signal_source_interface.h . . . 1644
telemetry_decoder_interface.h
This class represents an interface to a telemetry decoder block . . . 1645
tracking_interface.h
This class represents an interface to a tracking block . . . 1646
channel_event.h
Class that defines a channel event . . . 1647
channel_status_msg_receiver.h
GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks . . . 1649
command_event.h
Class that defines a receiver command event . . . 1650
galileo_e6_has_msg_receiver.h
GNU Radio block that processes Galileo HAS message pages received from Galileo E6B telemetry blocks. After successful decoding, sends the content to the PVT block . . . 1651
galileo_tow_map.h
GNU Radio block that stores TOW for Galileo channels . . . 1654
gnss_crypto.h
Class for computing cryptographic functions . . . 1656
gnss_sdr_fpga_sample_counter.h
Simple block to report the current receiver time based on the output of the tracking or telemetry blocks . . . 1658
gnss_sdr_log_sink.h
Buffered, per-run file logging for Abseil . . . 1659
gnss_sdr_sample_counter.h
Simple block to report the current receiver time based on the output of the tracking or telemetry blocks . . . 1661
gnss_sdr_supl_client.h
Class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library . . . 1663

gnss_sdr_time_counter.h
Simple block to report the current receiver time based on the output of the tracking or telemetry blocks . . . 1667
ini.h
This function parses an INI file into easy-to-access name/value pains . . . 1668
INIReader.h
This class reads an INI file into easy-to-access name/value pairs . . . 1670
nav_message_monitor.h
GNU Radio block that processes Nav_Message_Packet received from telemetry blocks and sends them via UDP . . . 1671
nav_message_packet.h
Class for storage of decoded navigation messages . . . 1673
nav_message_udp_sink.h
Interface of a class that sends serialized Nav_Message_Packet objects over UDP to one or multiple endpoints . . . 1674
osnma_crypto_material.h
OSNMA cryptographic material metadata and local cache manager . . . 1676
osnma_helper.h
Class for auxiliary osnma functions . . . 1677
osnma_msg_receiver.h
GNU Radio block that processes Galileo OSNMA data received from Galileo E1B telemetry blocks. After successful decoding, sends the content to the PVT block . . . 1679
osnma_nav_data_manager.h
Class for Galileo OSNMA navigation data management . . . 1685
protobuf_cleanup_manager.h
Singleton class that implements the shared resource cleaner pattern for the Protocol Buffers library . . . 1686
serdes_nav_message.h
Serialization / Deserialization of Nav_Message_Packet objects using Protocol Buffers . . . 1687
string_converter.h
Interface of a class that interprets the contents of a string and converts it into different types . . . 1689
uio_fpga.h
This library contains functions to determine the uio device driver file that corresponds to a hardware accelerator device name in the FPGA . . . 1690
gnss_synchro_monitor.h
Interface of a receiver monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP . . . 1691
gnss_synchro_udp_sink.h
Interface of a class that sends serialized Gnss_Synchro objects over udp to one or multiple endponits . . . 1693
serdes_gnss_synchro.h
Serialization / Deserialization of Gnss_Synchro objects using Protocol Buffers . . . 1694
concurrent_map.h
Interface of a thread-safe std::map . . . 1697
concurrent_queue.h
Interface of a thread-safe std::queue . . . 1699
control_thread.h
Interface of the receiver control plane . . . 1701
file_configuration.h
A ConfigurationInterface that reads the configuration from a file . . . 1704
gnss_block_factory.h
Interface of a factory that returns smart pointers to GNSS blocks . . . 1706
gnss_flowgraph.h
Interface of a GNSS receiver flow graph . . . 1708
in_memory_configuration.h
A ConfigurationInterface for testing purposes . . . 1713
satellite_visibility.h
Runtime satellite-visibility tracking for acquisition search prioritization . . . 1715

tcp_cmd_interface.h
Class that implements a TCP/IP telecommand command line interface for GNSS-SDR 1719
agnss_ref_location.h
Interface of an Assisted GNSS REFERENCE LOCATION storage 1720
agnss_ref_time.h
Interface of an Assisted GNSS REFERENCE TIME storage 1722
Beidou_B1C.h
Defines system parameters for BeiDou B1C open service signal 1724
Beidou_B1C_codes.h
Precomputed BeiDou B1C primary and secondary codes (declarations) 1726
Beidou_B1C_prn.h
B1C PRN Weil code parameters from BDS-SIS-ICD-B1C-1.0 Table 5-2/5-3/5-4 1727
Beidou_B1I.h
Defines system parameters for BeiDou B1I signal and DNAV data 1732
Beidou_B2a.h
System parameters for BeiDou B2a (BDS-SIS-ICD-B2a-1.0) 1734
Beidou_B3I.h
Defines system parameters for BeiDou B3I signal and DNAV data 1735
beidou_bdgim.h
BeiDou Global Ionospheric delay Model (BDGIM), BDS-SIS-ICD-B1C-1.0 §7.8 1737
Beidou_CNAV1.h
B-CNAV1 navigation message constants (BDS-SIS-ICD-B1C-1.0 §6.2, §7) 1738
beidou_cnav1_ephemeris.h
BeiDou B-CNAV1 ephemeris storage (BDS-SIS-ICD-B1C-1.0 §7) 1740
beidou_cnav1_iono.h
BeiDou B-CNAV1 ionospheric model (§7.8) 1742
beidou_cnav1_ldpc.h
B-CNAV1 NB-LDPC decoder (BDS-SIS-ICD-B1C-1.0 §6.2.2, Appendix) 1743
beidou_cnav1_navigation_message.h
B-CNAV1 navigation message parser (§6.2) 1745
beidou_cnav1_utc_model.h
BeiDou B-CNAV1 UTC model (BDS-SIS-ICD-B1C-1.0 §7.12) 1748
Beidou_CNAV2.h
B-CNAV2 navigation message constants (BDS-SIS-ICD-B2a-1.0) 1749
beidou_cnav2_ldpc.h
B-CNAV2 LDPC(96,48) decoder (BDS-SIS-ICD-B2a-1.0 section 6.2.2) 1750
beidou_cnav2_navigation_message.h
B-CNAV2 frame parser (BDS-SIS-ICD-B2a-1.0) 1752
Beidou_DNAV.h
Defines system parameters for BeiDou DNAV data processing 1753
beidou_dnav_almanac.h
Interface of a Beidou DNAV Almanac storage 1762
beidou_dnav_differential_corrections.h
Storage for BeiDou D2 integrity and differential corrections 1763
beidou_dnav_ephemeris.h
Interface of a BEIDOU EPHEMERIS storage 1765
beidou_dnav_iono.h
Interface of a BEIDOU IONOSPHERIC MODEL storage 1767
beidou_dnav_navigation_message.h
Interface of a BeiDou DNAV Data message decoder 1768
beidou_dnav_utc_model.h
Interface of a BeiDou UTC MODEL storage 1774
beidou_ldpc.h
Shared BeiDou B-CNAV1/B-CNAV2 non-binary LDPC decoder over GF(64) 1775
display.h
Defines useful display constants 1779
galileo_almanac.h
Interface of a Galileo ALMANAC storage 1781

galileo_almanac_helper.h
Interface of a Galileo ALMANAC storage helper . . . 1782
Galileo_CNAV.h
Galileo CNAV message constants. Data from: Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022 . . . 1784
galileo_cnav_message.h
Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022 . . . 1788
Galileo_E1.h
Defines system parameters for Galileo E1 signal and NAV data . . . 1790
Galileo_E5a.h
Defines system parameters for Galileo E5a signal and NAV data . . . 1808
Galileo_E5b.h
Defines system parameters for Galileo E5b signal and NAV data . . . 1852
Galileo_E6.h
Defines system parameters for Galileo E6 B/C signal, as published at: European Union, E6-B/C Codes Technical Note, Issue 1, January 2019 . . . 1895
galileo_ephemeris.h
Interface of a Galileo EPHEMERIS storage . . . 1918
Galileo_FNAV.h
Galileo FNAV message constants . . . 1921
galileo_fnav_message.h
Implementation of a Galileo F/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015) . . . 1926
galileo_has_data.h
Class for Galileo HAS message type 1 data storage . . . 1930
galileo_has_page.h
Class for Galileo HAS message page storage . . . 1933
Galileo_INAV.h
Galileo INAV message constants . . . 1934
galileo_inav_message.h
Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 2.0 (Jan. 2021) . . . 1943
galileo_iono.h
Interface of a Galileo Ionospheric Model storage . . . 1950
galileo_ism.h
Interface of a Galileo Integrity Support Message . . . 1952
Galileo_OSNMA.h
Galileo OSNMA message constants . . . 1955
galileo_reduced_ced.h
Galileo Reduced Clock and Ephemeris Data storage class . . . 1958
galileo_utc_model.h
Interface of a Galileo UTC MODEL storage . . . 1960
glonass_gnav_almanac.h
Interface of a GLONASS GNAV ALMANAC storage . . . 1961
glonass_gnav_ephemeris.h
Interface of a GLONASS EPHEMERIS storage . . . 1964
glonass_gnav_navigation_message.h
Interface of a GLONASS GNAV Data message decoder as described in GLONASS ICD (Edition 5.1) . . . 1967
glonass_gnav_utc_model.h
Interface of a GLONASS GNAV UTC MODEL storage . . . 1971
GLONASS_L1_L2_CA.h
Defines system parameters for GLONASS L1 C/A signal and NAV data . . . 1973
gnss_almanac.h
Base class for GNSS almanac storage . . . 1982
gnss_ephemeris.h
Base class for GNSS Ephemeris . . . 1983

gnss_frequencies.h
GNSS Frequencies . . . 1985
gnss_obs_codes.h
GNSS Observable codes . . . 1987
gnss_satellite.h
Interface of the Gnss_Satellite class . . . 1991
gnss_signal.h
Implementation of the Gnss_Signal class . . . 1993
gnss_synchro.h
Interface of the Gnss_Synchro class . . . 1994
gps_acq_assist.h
Interface of a GPS RRLL ACQUISITION ASSISTACE storage . . . 1997
gps_almanac.h
Interface of a GPS ALMANAC storage . . . 1999
GPS_CNAV.h
Defines parameters for GPS CNAV . . . 2000
gps_cnav_eop.h
Storage for GPS/QZSS CNAV Earth orientation parameters . . . 2005
gps_cnav_ephemeris.h
Interface of a GPS CNAV EPHEMERIS storage . . . 2006
gps_cnav_iono.h
Interface of a GPS CNAV IONOSPHERIC MODEL storage . . . 2009
gps_cnav_navigation_message.h
Interface of a GPS CNAV Data message decoder . . . 2010
gps_cnav_utc_model.h
Interface of a GPS CNAV UTC MODEL storage . . . 2012
gps_ephemeris.h
Interface of a GPS EPHEMERIS storage . . . 2013
gps_iono.h
Interface of a GPS IONOSPHERIC MODEL storage . . . 2015
GPS_L1_CA.h
Defines system parameters for GPS L1 C/A signal and NAV data . . . 2017
GPS_L2C.h
Defines system parameters for GPS L2C signal . . . 2025
GPS_L5.h
Defines system parameters for GPS L5 signal . . . 2027
gps_navigation_message.h
Interface of a GPS NAV Data message decoder . . . 2031
gps_utc_model.h
Interface of a GPS UTC MODEL storage . . . 2035
gps_week_rollover.h
Resolution of the GPS mod-1024 week-number rollover from the receiver configuration . . . 2037
MATH_CONSTANTS.h
Defines useful mathematical constants and their scaled versions . . . 2038
osnma_data.h
Class for Galileo OSNMA data storage . . . 2043
osnma_dsm_reader.h
Class for reading OSNMA DSM messages . . . 2048
qzss.h
Defines system parameters for QZSS signals . . . 2050
qzss_cnav_eop.h
Storage for QZSS CNAV Earth orientation parameters . . . 2051
qzss_cnav_iono.h
Interface of a QZSS CNAV ionospheric model storage . . . 2052
qzss_cnav_utc_model.h
Interface of a QZSS CNAV UTC model storage . . . 2053
qzss_iono.h
Interface of a QZSS ionospheric model storage . . . 2054

qzss_utc_model.h
Interface of a QZSS UTC model storage 2055
reed_solomon.h
Class implementing a Reed-Solomon encoder/decoder for blocks of 255 symbols and symbols of 8 bits 2056
sbas_ephemeris.h
Interface of a SBAS REFERENCE LOCATION storage 2058
SBAS_L1.h
Defines system parameters for SBAS L1 signal 2060
sbas_raw_message.h
Container for a CRC-validated raw SBAS L1 message 2062
signal_flag.h
Enum for signals 2063
tow_to_trk.h
Class to inform about TOW from Telemetry to Tracking blocks 2064

Chapter 10

Topic Documentation

10.1 Acquisition

Topics

- acquisition_adapters
- acquisition_gr_blocks
- acquisition_libs

10.1.1 Detailed Description

Classes for GNSS signal acquisition

Classes for GNSS signal acquisition

Classes for GNSS signal acquisition

10.1.2 acquisition_adapters

Classes

- class PcpsAcquisitionAdapter

  This class adapts a PCPS acquisition block to an AcquisitionInterface.
- class PcpsAcquisitionAdapterCustom

  This class adapts a PCPS acquisition block to an AcquisitionInterface.

10.1.2.1 Detailed Description

Wrap GNU Radio acquisition blocks with an AcquisitionInterface

Wrap GNU Radio acquisition blocks with an AcquisitionInterface

10.1.3 acquisition_gr_blocks

Classes

- class galileo_e5a_noncoherentIQ_acquisition_caf_cc

  This class implements a Parallel Code Phase Search Acquisition.
- class galileo_pcps_8ms_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).
- class pcps_acquisition

  This class implements a Parallel Code Phase Search Acquisition.
- class pcps_acquisition_fine_doppler_cc

  This class implements a Parallel Code Phase Search Acquisition.
- class pcps_acquisition_fpga

  This class implements a Parallel Code Phase Search Acquisition that uses the FPGA.
- class pcps_assisted_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition.
- class pcps_cccwsr_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme.
- class pcps_opencl_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition.
- class pcps_quicksync_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm.
- class pcps_tong_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.

Typedefs

- using galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr = gnss_shared_ptr<galileo_e5a_noncoherentIQ_acquisition_caf_c
- using galileo_pcps_8ms_acquisition_cc_sptr = gnss_shared_ptr<galileo_pcps_8ms_acquisition_cc>
- using pcps_acquisition_sptr = gnss_shared_ptr<pcps_acquisition>
- using pcps_acquisition_fine_doppler_cc_sptr = gnss_shared_ptr<pcps_acquisition_fine_doppler_cc>
- using pcps_acquisition_fpga_sptr = std::shared_ptr<pcps_acquisition_fpga>
- using pcps_assisted_acquisition_cc_sptr = gnss_shared_ptr<pcps_assisted_acquisition_cc>
- using pcps_cccwsr_acquisition_cc_sptr = gnss_shared_ptr<pcps_cccwsr_acquisition_cc>
- using pcps_opencl_acquisition_cc_sptr = gnss_shared_ptr<pcps_opencl_acquisition_cc>
- using pcps_quicksync_acquisition_cc_sptr = gnss_shared_ptr<pcps_quicksync_acquisition_cc>
- using pcps_tong_acquisition_cc_sptr = gnss_shared_ptr<pcps_tong_acquisition_cc>

Functions

- galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr **galileo_e5a_noncoherentIQ_make_acquisition_caf_cc** (const Acq_Conf &conf, bool both_signal_components, int CAF_window_hz, int Zero_padding)
- galileo_pcps_8ms_acquisition_cc_sptr **galileo_pcps_8ms_make_acquisition_cc** (const Acq_Conf &conf)
- pcps_acquisition_sptr **pcps_make_acquisition** (const Acq_Conf &conf_)
- pcps_acquisition_fine_doppler_cc_sptr **pcps_make_acquisition_fine_doppler_cc** (const Acq_Conf &conf_)
- pcps_acquisition_fpga_sptr **pcps_make_acquisition_fpga** (Acq_Conf_Fpga *conf_, uint32_t acq_buff_num, std::vector< std::pair< uint32_t, uint32_t > > &downsampling_filter_specs, uint32_t &max_FFT_size)
- pcps_assisted_acquisition_cc_sptr **pcps_make_assisted_acquisition_cc** (const Acq_Conf &conf)

- pcps_cccwsr_acquisition_cc_sptr **pcps_cccwsr_make_acquisition_cc** (const Acq_Conf &conf)
- pcps_opencl_acquisition_cc_sptr **pcps_make_opencl_acquisition_cc** (const Acq_Conf &conf, uint32_t max_dwells)
- pcps_quicksync_acquisition_cc_sptr **pcps_quicksync_make_acquisition_cc** (const Acq_Conf &conf, uint32_t folding_factor, uint32_t max_dwells)
- pcps_tong_acquisition_cc_sptr **pcps_tong_make_acquisition_cc** (const Acq_Conf &conf, uint32_t tong_init_val, uint32_t tong_max_val, uint32_t tong_max_dwells)

10.1.3.1 Detailed Description

GNU Radio processing blocks for GNSS signal acquisition

10.1.3.2 Typedef Documentation

10.1.3.2.1 galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr

```
using galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr = gnss_shared_ptr<galileo_e5a_noncoherentIQ_acquisiti
```

Definition at line 48 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

10.1.3.2.2 galileo_pcps_8ms_acquisition_cc_sptr

```
using galileo_pcps_8ms_acquisition_cc_sptr = gnss_shared_ptr<galileo_pcps_8ms_acquisition_cc>
```

Definition at line 42 of file galileo_pcps_8ms_acquisition_cc.h.

10.1.3.2.3 pcps_acquisition_fine_doppler_cc_sptr

```
using pcps_acquisition_fine_doppler_cc_sptr = gnss_shared_ptr<pcps_acquisition_fine_doppler_cc>
```

Definition at line 66 of file pcps_acquisition_fine_doppler_cc.h.

10.1.3.2.4 pcps_acquisition_fpga_sptr

```
using pcps_acquisition_fpga_sptr = std::shared_ptr<pcps_acquisition_fpga>
```

Definition at line 48 of file pcps_acquisition_fpga.h.

10.1.3.2.5 pcps_acquisition_sptr

```
using pcps_acquisition_sptr = gnss_shared_ptr<pcps_acquisition>
```

Definition at line 86 of file pcps_acquisition.h.

10.1.3.2.6 pcps_assisted_acquisition_cc_sptr

```
using pcps_assisted_acquisition_cc_sptr = gnss_shared_ptr<pcps_assisted_acquisition_cc>
```

Definition at line 58 of file pcps_assisted_acquisition_cc.h.

10.1.3.2.7 pcps_cccwsr_acquisition_cc_sptr

```
using pcps_cccwsr_acquisition_cc_sptr = gnss_shared_ptr<pcps_cccwsr_acquisition_cc>
```

Definition at line 47 of file pcps_cccwsr_acquisition_cc.h.

10.1.3.2.8 pcps_opencl_acquisition_cc_sptr

```
using pcps_opencl_acquisition_cc_sptr = gnss_shared_ptr<pcps_opencl_acquisition_cc>
```

Definition at line 65 of file pcps_opencl_acquisition_cc.h.

10.1.3.2.9 pcps_quicksync_acquisition_cc_sptr

```
using pcps_quicksync_acquisition_cc_sptr = gnss_shared_ptr<pcps_quicksync_acquisition_cc>
```

Definition at line 62 of file pcps_quicksync_acquisition_cc.h.

10.1.3.2.10 pcps_tong_acquisition_cc_sptr

```
using pcps_tong_acquisition_cc_sptr = gnss_shared_ptr<pcps_tong_acquisition_cc>
```

Definition at line 61 of file pcps_tong_acquisition_cc.h.

10.1.4 acquisition_libs

Classes

- class Acq_Conf
- class Acq_Conf_Fpga
- class CudaPcpsEngine

  Batched PCPS grid evaluation on a CUDA device.
- class Fpga_Acquisition

  Class that implements carrier wipe-off and correlators.

10.1.4.1 Detailed Description

Library with utilities for GNSS signal acquisition

Library with utilities for GNSS signal acquisition

10.2 Core GNSS Receiver

Header file of the interface to a signal_source GNSS block.

Topics

- GNSS block interfaces
- core_libs
- core_monitor
- core_receiver
- core_system_parameters

10.2.1 Detailed Description

Header file of the interface to a signal_source GNSS block.

\signal_source_interface.h

Author

Jim Melton, 2020. jim.melton(at)sncorp.com

This header file contains the interface to an abstract class for signal sources. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors)
GNSS-SDR is a software defined Global Navigation Satellite Systems receiver
This file is part of GNSS-SDR.
SPDX-License-Identifier: GPL-3.0-or-later

Core GNSS Receiver.

10.2.2 GNSS block interfaces

Classes

- class acquisition_impl_interface

  This abstract class represents an interface to an acquisition GNSS block.
- class Concurrent_Queue< Data >

  This class implements a thread-safe std::queue.
- class AcquisitionInterface

  This abstract class represents an interface to an acquisition GNSS block.
- class ChannelInterface

  This abstract class represents an interface to a channel GNSS block.
- class ConfigurationInterface

  This abstract class represents an interface to configuration parameters.
- class GNSSBlockInterface

  This abstract class represents an interface to GNSS blocks.
- class ObservablesInterface

  This abstract class represents an interface to an observables block.
- class PvtInterface

  This class represents an interface to a PVT block.
- class SignalSourceInterface

  This abstract class represents an interface to signal_source GNSS block.
- class TelemetryDecoderInterface

  This abstract class represents an interface to a navigation GNSS block.
- class TrackingInterface

  This abstract class represents an interface to a tracking block.

Typedefs

- using acquisition_impl_interface_sptr = gnss_shared_ptr<acquisition_impl_interface>
- template<typename T>
  using gnss_shared_ptr = boost::shared_ptr<T>

Functions

- template<typename C, typename... Args>
  gnss_shared_ptr< C > gnss_make_shared (Args &&... args)
- virtual size_t **SignalSourceInterface::getRfChannels** () const =0

10.2.2.1 Detailed Description

GNSS block interfaces.
GNSS block interfaces.
GNSS block interfaces.

10.2.2.2 Typedef Documentation

10.2.2.2.1 acquisition_impl_interface_sptr

```
using acquisition_impl_interface_sptr = gnss_shared_ptr<acquisition_impl_interface>
```
Definition at line 41 of file acquisition_impl_interface.h.

10.2.2.2.2 gnss_shared_ptr

```
template<typename T>
using gnss_shared_ptr = boost::shared_ptr<T>
```
Definition at line 50 of file gnss_block_interface.h.

10.2.2.3 Function Documentation

10.2.2.3.1 gnss_make_shared()

```
template<typename C, typename...  Args>
gnss_shared_ptr< C > gnss_make_shared (
            Args &&...  args)
```
Definition at line 52 of file gnss_block_interface.h.

10.2.2.3.2 SignalSourceInterface()

```
SignalSourceInterface::SignalSourceInterface ()  [inline], [protected]
```
Definition at line 57 of file signal_source_interface.h.

10.2.2.3.3 ~SignalSourceInterface()

```
SignalSourceInterface::~SignalSourceInterface ()  [inline]
```
Definition at line 63 of file signal_source_interface.h.

10.2.3 core_libs

Classes

- class Channel_Event
- class channel_status_msg_receiver

  GNU Radio block that receives asynchronous channel messages from tlm blocks.
- class Command_Event
- class galileo_e6_has_msg_receiver

GNU Radio block that receives asynchronous Galileo HAS message pages from the telemetry blocks, stores them in memory, and decodes HAS messages when enough data have been received. The decoded HAS message is sent to the PVT block.

- struct GalileoTowMapEntry
- class galileo_tow_map
- class Gnss_Crypto

  Class implementing cryptographic functions for Navigation Message Authentication.
- class gnss_sdr_fpga_sample_counter
- class GnssSdrLogSink
- class GnssSdrLogSinkGuard
- class gnss_sdr_sample_counter
- class Gnss_Sdr_Supl_Client

  class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library..
- class gnss_sdr_time_counter
- class INIReader

  Read an INI file into easy-to-access name/value pairs. (Note that I've gone for simplicity here rather than speed, but it should be pretty decent.).
- class nav_message_monitor

  GNU Radio block that receives asynchronous Nav_Message_Packet obkects from the telemetry blocks and sends them via UDP.
- class Nav_Message_Packet
- class Nav_Message_Udp_Sink
- class Osnma_Helper
- class osnma_msg_receiver

  GNU Radio block that receives asynchronous OSNMA messages from the telemetry blocks, stores them in memory, and decodes OSNMA info when enough data have been received. The decoded OSNMA data is sent to the PVT block.
- class OSNMA_NavDataManager

  Class for managing OSNMA navigation data.
- class Protobuf_Cleanup_Manager

  This class implements the Protocol Buffers library shared memory cleanup manager.
- class StringConverter

  Class that interprets the contents of a string and converts it into different types.

Macros

- #define INI_ALLOW_MULTILINE 1

Typedefs

- using channel_event_sptr = std::shared_ptr<Channel_Event>
- using channel_status_msg_receiver_sptr = gnss_shared_ptr<channel_status_msg_receiver>
- using command_event_sptr = std::shared_ptr<Command_Event>
- using galileo_e6_has_msg_receiver_sptr = gnss_shared_ptr<galileo_e6_has_msg_receiver>
- using galileo_tow_map_sptr = gnss_shared_ptr<galileo_tow_map>
- using GalileoTowMap = std::map<uint32_t, GalileoTowMapEntry>
- using GalileoTowMapMessage = std::pair<uint32_t, GalileoTowMapEntry>
- using gnss_sdr_fpga_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_fpga_sample_counter>
- using gnss_sdr_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_sample_counter>
- using gnss_sdr_time_counter_sptr = std::shared_ptr<gnss_sdr_time_counter>
- using nav_message_monitor_sptr = gnss_shared_ptr<nav_message_monitor>
- using b_io_context = boost::asio::io_service
- using osnma_msg_receiver_sptr = gnss_shared_ptr<osnma_msg_receiver>

Functions

- channel_event_sptr **channel_event_make** (int channel_id, int event_type)
- channel_status_msg_receiver_sptr **channel_status_msg_receiver_make** ()
- command_event_sptr **command_event_make** (int command_id, int event_type)
- galileo_e6_has_msg_receiver_sptr **galileo_e6_has_msg_receiver_make** ()
- galileo_tow_map_sptr **galileo_tow_map_make** ()
- gnss_sdr_fpga_sample_counter_sptr **gnss_sdr_make_fpga_sample_counter** (double _fs, int32_t _interval_ms)
- gnss_sdr_sample_counter_sptr **gnss_sdr_make_sample_counter** (double _fs, int32_t _interval_ms, size_t _size)
- gnss_sdr_time_counter_sptr **gnss_sdr_make_time_counter** ()
- int ini_parse (const char *filename, int(*handler)(void *user, const char *section, const char *name, const char *value), void *user)

  Parse given INI-style file. May have [section]s, name=value pains (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pain parsed before any section heading.
- nav_message_monitor_sptr **nav_message_monitor_make** (const std::vector< std::string > &addresses, uint16_t port)
- osnma_msg_receiver_sptr **osnma_msg_receiver_make** (const std::string &pemFilePath, const std::string &merkleFilePath, bool strict_mode=false, bool replay_mode=false)
- const std::string **uio_dir** ("/sys/class/uio/")
- const std::string **uio_filename** ("uio")
- const std::string **uio_subdir_name** ("/name")
- int32_t find_uio_dev_file_name (std::string &device_file_name, const std::string &device_name, uint32_t device_num)

  This function finds the uio device driver device file name out of the device name and the device number.
- uint32_t get_num_devices (const std::string &device_name)

  Returns the number of devices (tracking multicorrelators) of type device_name available in the FPGA.

Variables

- constexpr uint32_t GALILEO_TOW_MAP_WEEK_MS = 604800000U
- constexpr uint32_t GALILEO_TOW_MAP_INVALID_WEEK = std::numeric_limits<uint32_t>::max()
- constexpr uint32_t GALILEO_TOW_MAP_INVALID_TOW_MS = std::numeric_limits<uint32_t>::max()
- constexpr uint64_t GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER = std::numeric_limits<uint64_t>::max()

10.2.3.1 Detailed Description

Utilities for the core GNSS receiver.
Utilities for the core GNSS receiver.

10.2.3.2 Macro Definition Documentation

10.2.3.2.1 INI_ALLOW_MULTILINE

```
#define INI_ALLOW_MULTILINE 1
```

Definition at line 65 of file ini.h.

10.2.3.3 Typedef Documentation

10.2.3.3.1 b_io_context

```
using b_io_context = boost::asio::io_service
```

Definition at line 36 of file nav_message_udp_sink.h.

10.2.3.3.2 channel_event_sptr

```
using channel_event_sptr = std::shared_ptr<Channel_Event>
```
Definition at line 30 of file channel_event.h.

10.2.3.3.3 channel_status_msg_receiver_sptr

```
using channel_status_msg_receiver_sptr = gnss_shared_ptr<channel_status_msg_receiver>
```
Definition at line 37 of file channel_status_msg_receiver.h.

10.2.3.3.4 command_event_sptr

```
using command_event_sptr = std::shared_ptr<Command_Event>
```
Definition at line 30 of file command_event.h.

10.2.3.3.5 galileo_e6_has_msg_receiver_sptr

```
using galileo_e6_has_msg_receiver_sptr = gnss_shared_ptr<galileo_e6_has_msg_receiver>
```
Definition at line 46 of file galileo_e6_has_msg_receiver.h.

10.2.3.3.6 galileo_tow_map_sptr

```
using galileo_tow_map_sptr = gnss_shared_ptr<galileo_tow_map>
```
Definition at line 35 of file galileo_tow_map.h.

10.2.3.3.7 GalileoTowMap

```
using GalileoTowMap = std::map<uint32_t, GalileoTowMapEntry>
```
Definition at line 59 of file galileo_tow_map.h.

10.2.3.3.8 GalileoTowMapMessage

```
using GalileoTowMapMessage = std::pair<uint32_t, GalileoTowMapEntry>
```
Definition at line 60 of file galileo_tow_map.h.

10.2.3.3.9 gnss_sdr_fpga_sample_counter_sptr

```
using gnss_sdr_fpga_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_fpga_sample_counter>
```
Definition at line 36 of file gnss_sdr_fpga_sample_counter.h.

10.2.3.3.10 gnss_sdr_sample_counter_sptr

```
using gnss_sdr_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_sample_counter>
```
Definition at line 36 of file gnss_sdr_sample_counter.h.

10.2.3.3.11 gnss_sdr_time_counter_sptr

```
using gnss_sdr_time_counter_sptr = std::shared_ptr<gnss_sdr_time_counter>
```
Definition at line 34 of file gnss_sdr_time_counter.h.

10.2.3.3.12 nav_message_monitor_sptr

```
using nav_message_monitor_sptr = gnss_shared_ptr<nav_message_monitor>
```
Definition at line 37 of file nav_message_monitor.h.

10.2.3.3.13 osnma_msg_receiver_sptr

```
using osnma_msg_receiver_sptr = gnss_shared_ptr<osnma_msg_receiver>
```
Definition at line 54 of file osnma_msg_receiver.h.

10.2.3.4 Function Documentation

10.2.3.4.1 find_uio_dev_file_name()

```
int32_t find_uio_dev_file_name (
            std::string & device_file_name,
            const std::string & device_name,
            uint32_t device_num)
```

This function finds the uio device driver device file name out of the device name and the device number.
References find_uio_dev_file_name().
Referenced by find_uio_dev_file_name().

10.2.3.4.2 get_num_devices()

```
uint32_t get_num_devices (
            const std::string & device_name)
```

Returns the number of devices (tracking multicorrelators) of type device_name available in the FPGA.
References get_num_devices().
Referenced by get_num_devices().

10.2.3.4.3 ini_parse()

```
int ini_parse (
            const char * filename,
            int(* handler )(void *user, const char *section, const char *name, const char
*value),
            void * user)
```

Parse given INI-style file. May have [section]s, name=value pains (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pain parsed before any section heading.
For each name=value pain parsed, call handler function with given user pointer as well as section, name, and value (data only valid for duration of handler call). Handler should return nonzero on success, zero on error.
Returns 0 on success, line number of first error on parse error, on -1 on file open error.
References ini_parse().
Referenced by ini_parse().

10.2.3.5 Variable Documentation

10.2.3.5.1 GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER

```
uint64_t GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER = std::numeric_limits<uint64_t>::max()  [constexpr]
```

Definition at line 42 of file galileo_tow_map.h.

10.2.3.5.2 GALILEO_TOW_MAP_INVALID_TOW_MS

```
uint32_t GALILEO_TOW_MAP_INVALID_TOW_MS = std::numeric_limits<uint32_t>::max()  [constexpr]
```

Definition at line 41 of file galileo_tow_map.h.

10.2.3.5.3 GALILEO_TOW_MAP_INVALID_WEEK

```
uint32_t GALILEO_TOW_MAP_INVALID_WEEK = std::numeric_limits<uint32_t>::max()  [constexpr]
```

Definition at line 40 of file galileo_tow_map.h.

10.2.3.5.4 GALILEO_TOW_MAP_WEEK_MS

```
uint32_t GALILEO_TOW_MAP_WEEK_MS = 604800000U  [constexpr]
```

Definition at line 39 of file galileo_tow_map.h.

10.2.4 core_monitor

Classes

- class gnss_synchro_monitor

  This class implements a monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.
- class Gnss_Synchro_Udp_Sink

  This class sends serialized Gnss_Synchro objects over UDP to one or multiple endpoints.

Typedefs

- using gnss_synchro_monitor_sptr = gnss_shared_ptr<gnss_synchro_monitor>
- using b_io_context = boost::asio::io_service

Functions

- gnss_synchro_monitor_sptr **gnss_synchro_make_monitor** (int n_channels, int decimation_factor, const std::vector< std::string > &udp_ports, const std::vector< std::string > &udp_addresses, bool enable_↩ protobuf)

10.2.4.1 Detailed Description

Classes for the Gnss_Synchro monitor.

10.2.4.2 Typedef Documentation

10.2.4.2.1 b_io_context

```
using b_io_context = boost::asio::io_service
```

Definition at line 38 of file gnss_synchro_udp_sink.h.

10.2.4.2.2 gnss_synchro_monitor_sptr

```
using gnss_synchro_monitor_sptr = gnss_shared_ptr<gnss_synchro_monitor>
```

Definition at line 40 of file gnss_synchro_monitor.h.

10.2.5 core_receiver

Namespaces

- namespace block_factory

  Class that produces all kinds of GNSS blocks.

Classes

- class Concurrent_Map< Data >

  This class implements a thread-safe std::map.
- class Concurrent_Queue< Data >

  This class implements a thread-safe std::queue.
- class ControlThread

  This class represents the main thread of the application, so the name is ControlThread. This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions.
- class FileConfiguration

  This class is an implementation of the interface ConfigurationInterface.
- class GNSSFlowgraph

  This class represents a GNSS flow graph.
- class InMemoryConfiguration

This class is an implementation of the interface ConfigurationInterface.

- class SatelliteVisibility

  Opt-in acquisition search classification: visible, excluded, or unknown. Healthy satellites above the search mask are preferred; unknowns are searched less often. Banned PRNs are omitted, and tracking can override exclusion. Tick() refreshes on reference, fix, navigation-data, position, interval, or expiry changes. Expired data returns satellites to the unknown pool. Disabled by default (GNSS-SDR.enable_visibility_aware_search).

- class TcpCmdInterface

Functions

- std::vector< std::pair< int, Gnss_Satellite > > compute_visible_satellites (const std::shared_ptr< PvtInterface > &pvt_ptr, const gtime_t &gps_gtime, const arma::vec &r_eb_e, double elevation_mask_deg, std::vector< std::pair< int, Gnss_Satellite > > *below_mask_out=nullptr, double almanac_max_age_s=std::numeric_limits< double >::infinity(), double *seconds_until_next_expiry_out=nullptr, const std::set< std::pair< std::string, uint32_t > > *only_prns=nullptr, bool glonass_strict_health=true)

  Return healthy GPS/Galileo/BeiDou/GLONASS/QZSS satellites above the mask. Prefer fresh ephemeris over almanac. BeiDou selects the closest orbit (ties: DNAV, CNAV1, CNAV2); B-CNAV supports IGSO/MEO only. Unhealthy satellites are excluded, not unknown; Galileo uses E1B health.

10.2.5.1 Detailed Description

Classes for the core GNSS receiver.

10.2.5.2 Function Documentation

10.2.5.2.1 compute_visible_satellites()

```
std::vector< std::pair< int, Gnss_Satellite > > compute_visible_satellites (
            const std::shared_ptr< PvtInterface > & pvt_ptr,
            const gtime_t & gps_gtime,
            const arma::vec & r_eb_e,
            double elevation_mask_deg,
            std::vector< std::pair< int, Gnss_Satellite > > * below_mask_out = nullptr,
            double almanac_max_age_s = std::numeric_limits< double >::infinity(),
            double * seconds_until_next_expiry_out = nullptr,
            const std::set< std::pair< std::string, uint32_t > > * only_prns = nullptr,
            bool glonass_strict_health = true)
```

Return healthy GPS/Galileo/BeiDou/GLONASS/QZSS satellites above the mask. Prefer fresh ephemeris over almanac. BeiDou selects the closest orbit (ties: DNAV, CNAV1, CNAV2); B-CNAV supports IGSO/MEO only. Unhealthy satellites are excluded, not unknown; Galileo uses E1B health.

Parameters

gps_gtime	GPST query epoch; also resolves truncated navigation weeks.
r_eb_e	receiver ECEF position (m).
elevation_mask_deg	search mask (deg), independent of PVT.elevation_mask.
below_mask_out	optional output for satellites at/below the mask or unhealthy.
almanac_max_age_s	maximum absolute almanac age (s); infinity disables it. Ephemeris age limits use RTKLIB's per-system MAXDTOE* constants.
seconds_until_next_expiry_out	optional time to earliest classified-data expiry; infinity if nothing was classified.
only_prns	optional (system, PRN) filter; requires unchanged position/time.
glonass_strict_health	also reject the Bn MSB; In always marks unhealthy.

Returns

(floor(elevation) in degrees, satellite) pairs, highest elevation first.

References compute_visible_satellites().
Referenced by compute_visible_satellites().

10.2.6 core_system_parameters

Classes

- class Agnss_Ref_Location

  Interface of an Assisted GNSS REFERENCE LOCATION storage.
- class Agnss_Ref_Time

  Interface of an Assisted GNSS REFERENCE TIME storage.
- class Beidou_Cnav1_Iono
- struct Bds3_B1c_PageCommon
- struct Bds3_B1c_AlmanacReduced
- struct Bds3_B1c_AlmanacMedium
- struct Bds3_B1c_Eop
- struct Bds3_B1c_Bgto
- struct Bds3_B1c_PageData
- struct Beidou_Cnav1_PageData_Message
- class Beidou_Cnav1_Navigation_Message
- class Beidou_Cnav1_Utc_Model
- class Beidou_Cnav2_Navigation_Message
- class Beidou_Dnav_Almanac

  This class is a storage for the BeiDou D1/D2 almanac.
- struct Beidou_Dnav_Differential_Correction
- class Beidou_Dnav_Differential_Corrections
- class Beidou_Dnav_Ephemeris

  This is a storage class for the Beidou SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0).
- class Beidou_Dnav_Iono

  This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1.
- class Beidou_Dnav_Navigation_Message

  This class decodes BeiDou D1 and D2 NAV data messages.
- class Beidou_Dnav_Utc_Model

  This class is a storage for the BeiDou DNAV UTC Model.
- class Galileo_Almanac

  This class is a storage for the Galileo SV ALMANAC data.
- class Galileo_Almanac_Helper

  This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD.
- class Galileo_Cnav_Message

  This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
- class Galileo_Ephemeris
- class Galileo_Fnav_Message

  This class handles the Galileo F/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
- struct mt1_header
- class Galileo_HAS_data

  This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022. See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf.

- class Galileo_HAS_page

  This class is a storage for Galileo HAS message page, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
- class OSNMA_msg

  This class fills the OSNMA_msg structure with the data received from the telemetry blocks.
- class Galileo_Inav_Message

  This class handles the Galileo I/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
- class Galileo_Iono

  This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6.
- class Galileo_ISM

  This class is a storage for the GALILEO Integrity Support Message as described in Galileo ICD paragraph 5.2.
- class Mack_lookup
- class Galileo_Reduced_CED

  This class manages the Galileo Reduced Clock and Ephemeris Data.
- class Galileo_Utc_Model

  This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf paragraph 5.1.7.
- class Glonass_Gnav_Almanac

  This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1).
- class Glonass_Gnav_Ephemeris

  This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1).
- class Glonass_Gnav_Navigation_Message

  This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1).
- class Glonass_Gnav_Utc_Model

  This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1).
- class Gnss_Almanac

  Base class for GNSS almanac storage.
- class Gnss_Satellite

  This class represents a GNSS satellite.
- class Gnss_Signal

  This class represents a GNSS signal.
- class Gnss_Synchro

  This is the class that contains the information that is shared by the processing blocks.
- class Gps_Acq_Assist

  This class is a storage for the GPS GSM RRLL acquisition assistance data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5).
- class Gps_Almanac

  This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M.
- class Gps_CNAV_Eop

  Storage for the Earth orientation parameters broadcast in GPS/QZSS CNAV Message Type 32, as described in IS-GPS-200M.
- class Gps_CNAV_Ephemeris

  This is a storage class for the GPS CNAV ephemeris data as described in IS-GPS-200M.
- class Gps_CNAV_Iono

  This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
- class Gps_CNAV_Navigation_Message

  This class decodes a GPS CNAV Data message as described in IS-GPS-200M.
- class Gps_CNAV_Utc_Model

This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M.

- class Gps_Ephemeris

  This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200M.
- class Gps_Iono

  This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
- class Gps_Navigation_Message

  This class decodes a GPS NAV Data message as described in IS-GPS-200M.
- class Gps_Utc_Model

  This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M.
- class DSM_nma_header
- class DSM_dsm_header
- class MACK_header
- class MACK_tag_info
- class MACK_tag_and_info
- class DSM_PKR_message
- class DSM_KROOT_message
- class MACK_message
- class OSNMA_NavData
- class OSNMA_data

  This class handles ONSMA data See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf.
- class Tag
- class OSNMA_DSM_Reader
- class Qzss_CNAV_Eop
- class Qzss_CNAV_Iono

  This class is a storage for the QZSS CNAV Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
- class Qzss_CNAV_Utc_Model

  This class is a storage for the QZSS CNAV UTC model data as described in IS-QZSS-PNT-006.
- class Qzss_Iono

  This class is a storage for the QZSS Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
- class Qzss_Utc_Model

  This class is a storage for the QZSS UTC model data as described in IS-QZSS-PNT-006.
- class ReedSolomon

  Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where k=255-nroots is the information vector length and d=nroots+1 is the minimum Hamming distance, with symbols of 8 bits. It allows shortened RS codes.
- class Sbas_Ephemeris

  This class stores SBAS SV ephemeris data.
- class Sbas_Raw_Message

  A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp.
- class TOW_to_trk

Macros

- #define DISPLAY_COLORS 1
- #define GLONASS_GNAV_PREAMBLE
- #define GLONASS_GNAV_PREAMBLE_SAMPLES

Typedefs

- using BeidouCnav1LdpcGraph = BeidouLdpcGraph

Enumerations

- enum class Galileo_Nav_Message_Type : uint8_t { **Unknown** = 0 , **INAV** = 1 , **FNAV** = 2 }

  This class is a storage and orbital model functions for the Galileo SV ephemeris data as described in Galileo ICD paragraph 5.1.1.
- enum class **Galileo_Nav_Message_Source** : uint8_t { **Unknown** = 0 , **E1B** = 1 , **E5a** = 2 , **E5b** = 3 }
- enum class **CnavSystem** { **GPS** , **QZSS** }
- enum class **LnavSystem** { **GPS** , **QZSS** }

Functions

- Agnss_Ref_Location parse_agnss_ref_location (const std::string &ref_location_str)

  Parses GNSS-SDR.AGNSS_ref_location ("lat,lon" or "lat lon", degrees). valid is false if the string is empty or not a plausible WGS84 position. Single parsing point for ControlThread and SatelliteVisibility.
- Agnss_Ref_Time parse_agnss_ref_utc_time (const std::string &ref_time_str, bool *malformed_year=nullptr, bool *malformed_format=nullptr)

  Parses GNSS-SDR.AGNSS_ref_utc_time ("DD/MM/YYYY HH:MM:SS", UTC). An empty string is not an error: the host's current time is returned, marked valid. malformed_year and malformed_format (both optional) tell why a non-empty string was rejected. Single parsing point for ControlThread and SatelliteVisibility.
- double beidou_bdgim_delay_m (double time_mjd, double lat_rad, double lon_rad, double az_rad, double el_rad, const double alpha[9], double freq_hz)

  Compute BDGIM slant ionospheric delay [m] at carrier frequency freq_hz.
- double beidou_bdgim_nearest_odd_hour_mjd (double time_mjd)

  ICD §7.8.2: odd hour of day (01:00 . . . 23:00) nearest to time_mjd [MJD days].
- bool **beidou_cnav1_ldpc_decode_200_100** (const float *symbol_llr, int32_t num_bits, uint8_t *info_↩ bits600)
- bool **beidou_cnav1_ldpc_decode_88_44** (const float *symbol_llr, int32_t num_bits, uint8_t *info_bits264)
- bool **beidou_cnav1_ldpc_decode_200_100_codeword** (const float *symbol_llr, int32_t num_bits, uint8_t *codeword_bits1200)
- bool **beidou_cnav1_ldpc_decode_88_44_codeword** (const float *symbol_llr, int32_t num_bits, uint8_↩ t *codeword_bits528)
- bool **beidou_cnav1_ldpc_init_graph** (int32_t num_checks, int32_t num_variables, int32_t row_weight, const uint16_t *icd_index, const uint8_t *icd_element, int32_t num_entries, BeidouCnav1LdpcGraph &graph)
- const BeidouCnav1LdpcGraph & **beidou_cnav1_ldpc_graph_200_100** ()
- const BeidouCnav1LdpcGraph & **beidou_cnav1_ldpc_graph_88_44** ()
- const BeidouLdpcGraph & **beidou_cnav2_ldpc_graph_96_48** ()
- bool **beidou_cnav2_ldpc_decode_96_48** (const float *bit_llr, int32_t num_bits, uint8_t *info_bits288)
- bool **beidou_cnav2_ldpc_decode_96_48_codeword** (const float *bit_llr, int32_t num_bits, uint8_↩ t *codeword_bits576)
- const std::vector< std::pair< int32_t, int32_t > > **D1_PRE** ({{1, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_FRAID** ({{16, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SOW** ({{19, 8}, {31, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_PNUM** ({{44, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SAT_H1** ({{43, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AODC** ({{44, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_URAI** ({{49, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WN** ({{61, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOC** ({{74, 9}, {91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TGD1** ({{99, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TGD2** ({{109, 4}, {121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA0** ({{127, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA1** ({{135, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA2** ({{151, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA3** ({{159, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA0** ({{167, 6}, {181, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA1** ({{183, 8}})

- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA2** ({{191, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA3** ({{199, 4}, {211, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A2** ({{215, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0** ({{226, 7}, {241, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1** ({{258, 5}, {271, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AODE** ({{288, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_N** ({{43, 10}, {61, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CUC** ({{67, 16}, {91, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_M0** ({{93, 20}, {121, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_E** ({{133, 10}, {151, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CUS** ({{181, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CRC** ({{199, 4}, {211, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CRS** ({{225, 8}, {241, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SQRT_A** ({{251, 12}, {271, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOE_SF2** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOE_SF3** ({{43, 10}, {61, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_I0** ({{66, 17}, {91, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CIC** ({{106, 7}, {121, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_DOT** ({{132, 11}, {151, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CIS** ({{164, 9}, {181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_IDOT** ({{190, 13}, {211, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA0** ({{212, 21}, {241, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA** ({{252, 11}, {271, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SQRT_A_ALMANAC** ({{51, 2}, {61, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1_ALMANAC** ({{91, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0_ALMANAC** ({{102, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA0_ALMANAC** ({{121, 22}, {151, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_E_ALMANAC** ({{153, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_I** ({{170, 3}, {181, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOA** ({{194, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_DOT_ALMANAC** ({{202, 1}, {211, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_ALMANAC** ({{227, 6}, {241, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_M0_ALMANAC** ({{259, 4}, {271, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMEPID** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMID** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMID_HEALTH** ({{216, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA1** ({{51, 2}, {61, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA2** ({{68, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA3** ({{77, 6}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA4** ({{94, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA5** ({{103, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA6** ({{112, 1}, {121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA7** ({{129, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA8** ({{138, 5}, {151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA9** ({{155, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA10** ({{164, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA11** ({{181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA12** ({{190, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA13** ({{199, 4}, {211, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA14** ({{216, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA15** ({{225, 8}, {241, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA16** ({{242, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA17** ({{251, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA18** ({{260, 3}, {271, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA19** ({{277, 9}})

- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA20** ({{51, 2}, {61, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA21** ({{68, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA22** ({{77, 6}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA23** ({{94, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA24** ({{103, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA25** ({{112, 1}, {121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA26** ({{129, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA27** ({{138, 5}, {151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA28** ({{155, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA29** ({{164, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA30** ({{181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WNA** ({{190, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOA2** ({{198, 5}, {211, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GPS** ({{97, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GPS** ({{111, 2}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GAL** ({{135, 8}, {151, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GAL** ({{157, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GLO** ({{181, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GLO** ({{195, 8}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_T_LS** ({{51, 2}, {61, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_T_LSF** ({{67, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WN_LSF** ({{75, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0UTC** ({{91, 22}, {121, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1UTC** ({{131, 12}, {151, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DN** ({{163, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PRE** ({{1, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_FRAID** ({{16, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SOW** ({{19, 8}, {31, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM1** ({{43, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM** (D2_PNUM1)
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM2** ({{44, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM5** ({{44, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SAT_H1** ({{47, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_AODC** ({{48, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_URAI** ({{61, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_WN** ({{65, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TOC** ({{78, 5}, {91, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TGD1** ({{103, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TGD2** ({{121, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA0** ({{47, 6}, {61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA1** ({{63, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA2** ({{71, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA3** ({{79, 4}, {91, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA0** ({{95, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA1** ({{103, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA2** ({{111, 2}, {121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA3** ({{127, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A0** ({{101, 12}, {121, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1_MSB** ({{133, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1_LSB** ({{47, 6}, {61, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1** ({{279, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A2** ({{73, 10}, {91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_AODE** ({{92, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_N** ({{97, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC_MSB** ({{121, 14}})

- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC_LSB** ({{47, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC** ({{283, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_M0** ({{51, 2}, {61, 22}, {91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUS** ({{99, 14}, {121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_E_MSB** ({{125, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_E_LSB** ({{47, 6}, {61, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SQRT_A** ({{77, 6}, {91, 22}, {121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC_MSB** ({{125, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC_LSB** ({{47, 6}, {61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC** ({{283, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIS** ({{63, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TOE** ({{81, 2}, {91, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0_MSB** ({{106, 7}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0_LSB** ({{47, 6}, {61, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0** ({{269, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CRC** ({{66, 17}, {91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CRS** ({{92, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT_MSB** ({{110, 3}, {121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT_LSB** ({{47, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT** ({{277, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA0** ({{52, 1}, {61, 22}, {91, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_MSB** ({{100, 13}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_LSB** ({{47, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA** ({{269, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_IDOT** ({{52, 1}, {61, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SAT_H2** ({{48, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDID1_30** ({{50, 3}, {61, 22}, {91, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF2** ({{276, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF2** ({{280, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF3_1** ({{44, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF3_1** ({{48, 5}, {61, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF3_2** ({{69, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF3_2** ({{73, 10}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDEPID** ({{169, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDID31_63** ({{171, 2}, {181, 22}, {211, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF4** ({{252, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF4** ({{256, 7}, {271, 6}})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_STATUS** ({1, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_RESERVED** ({3, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_TYPE** ({5, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_ID** ({7, 5})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_SIZE** ({12, 5})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_PAGE_ID** ({17, 8})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_TOH** ({1, 12})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_MASK_FLAG** ({13, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_ORBIT_CORRECTION_FLAG** ({14, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CLOCK_FULLSET_FLAG** ({15, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CLOCK_SUBSET_FLAG** ({16, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CODE_BIAS_FLAG** ({17, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_PHASE_BIAS_FLAG** ({18, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_RESERVED** ({19, 4})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_MASK_ID** ({23, 5})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_IOD_SET_ID** ({28, 5})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_PAGE_TYPE_BIT** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SV_ID_PRN_1_BIT** ({{7, 6}})

- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_1_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0C_1_BIT** ({{23, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_1_BIT** ({{37, 31}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_1_BIT** ({{68, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF2_1_BIT** ({{89, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SISA_1_BIT** ({{95, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI0_1_BIT** ({{103, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI1_1_BIT** ({{114, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI2_1_BIT** ({{125, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION1_1_BIT** ({{139, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION2_1_BIT** ({{140, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION3_1_BIT** ({{141, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION4_1_BIT** ({{142, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION5_1_BIT** ({{143, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_BGD_1_BIT** ({{144, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_1_BIT** ({{154, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_1_BIT** ({{156, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_1_BIT** ({{168, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5ADVS_1_BIT** ({{188, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_2_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_2_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_2_BIT** ({{49, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_2_BIT** ({{73, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A12_2_BIT** ({{105, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_2_BIT** ({{137, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IDOT_2_BIT** ({{169, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_2_BIT** ({{183, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_2_BIT** ({{195, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_3_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_I0_3_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_3_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAN_3_BIT** ({{81, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CUC_3_BIT** ({{97, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CUS_3_BIT** ({{113, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CRC_3_BIT** ({{129, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CRS_3_BIT** ({{145, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0E_3_BIT** ({{161, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_3_BIT** ({{175, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_3_BIT** ({{187, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_4_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CIC_4_BIT** ({{17, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CIS_4_BIT** ({{33, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A0_4_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A1_4_BIT** ({{81, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTATLS_4_BIT** ({{105, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0T_4_BIT** ({{113, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NOT_4_BIT** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NLSF_4_BIT** ({{129, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DN_4_BIT** ({{137, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTATLSF_4_BIT** ({{140, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0G_4_BIT** ({{148, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A0G_4_BIT** ({{156, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A1G_4_BIT** ({{172, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_N0G_4_BIT** ({{184, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_4_BIT** ({{190, 20}})

- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DA_5_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NA_5_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0A_5_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D1_5_BIT** ({{23, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_1_5_BIT** ({{29, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_1_5_BIT** ({{42, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_1_5_BIT** ({{53, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_1_5_BIT** ({{69, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_1_5_BIT** ({{80, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_1_5_BIT** ({{96, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_1_5_BIT** ({{107, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_1_5_BIT** ({{123, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_1_5_BIT** ({{139, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_1_5_BIT** ({{152, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D2_5_BIT** ({{154, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_2_5_BIT** ({{160, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_2_5_BIT** ({{173, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_2_5_BIT** ({{184, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_2_5_BIT** ({{200, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DA_6_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_2_6_BIT** ({{23, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_2_6_BIT** ({{34, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_2_6_BIT** ({{50, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_2_6_BIT** ({{66, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_2_6_BIT** ({{79, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D3_6_BIT** ({{81, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_3_6_BIT** ({{87, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_3_6_BIT** ({{100, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_3_6_BIT** ({{111, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_3_6_BIT** ({{127, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_3_6_BIT** ({{138, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_3_6_BIT** ({{154, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_3_6_BIT** ({{165, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_3_6_BIT** ({{181, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_3_6_BIT** ({{197, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_3_6_BIT** ({{210, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **TYPE** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **PAGE_TYPE_BIT** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_1_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **T0_E_1_BIT** ({{17, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_1_BIT** ({{31, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **E_1_BIT** ({{63, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1_BIT** ({{95, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_2_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_0_2_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **I_0_2_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_2_BIT** ({{81, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **I_DOT_2_BIT** ({{113, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_3_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_3_BIT** ({{17, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_N_3_BIT** ({{41, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_UC_3_BIT** ({{57, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_US_3_BIT** ({{73, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RC_3_BIT** ({{89, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RS_3_BIT** ({{105, 16}})

- const std::vector< std::pair< int32_t, int32_t > > **SISA_3_BIT** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_4_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_ID_PRN_4_BIT** ({{17, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IC_4_BIT** ({{23, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IS_4_BIT** ({{39, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **T0C_4_BIT** ({{55, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_4_BIT** ({{69, 31}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_4_BIT** ({{100, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **AF2_4_BIT** ({{121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **SPARE_4_BIT** ({{127, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **AI0_5_BIT** ({{7, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **AI1_5_BIT** ({{18, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **AI2_5_BIT** ({{29, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION1_5_BIT** ({{43, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION2_5_BIT** ({{44, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION3_5_BIT** ({{45, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION4_5_BIT** ({{46, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION5_5_BIT** ({{47, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **BGD_E1_E5A_5_BIT** ({{48, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **BGD_E1_E5B_5_BIT** ({{58, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_5_BIT** ({{68, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_5_BIT** ({{70, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_DVS_5_BIT** ({{72, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_DVS_5_BIT** ({{73, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_5_BIT** ({{74, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_5_BIT** ({{86, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **SPARE_5_BIT** ({{106, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **A0_6_BIT** ({{7, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **A1_6_BIT** ({{39, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_LS_6_BIT** ({{63, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T0T_6_BIT** ({{71, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **W_NOT_6_BIT** ({{79, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_LSF_6_BIT** ({{87, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DN_6_BIT** ({{95, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_LSF_6_BIT** ({{98, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_6_BIT** ({{106, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_7_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A_7_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T0A_7_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D1_7_BIT** ({{23, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_7_BIT** ({{29, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_7_BIT** ({{42, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_7_BIT** ({{53, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_7_BIT** ({{69, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_7_BIT** ({{80, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_7_BIT** ({{96, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_7_BIT** ({{107, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_8_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_8_BIT** ({{11, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_8_BIT** ({{27, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_8_BIT** ({{40, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_8_BIT** ({{42, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D2_8_BIT** ({{44, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_8_BIT** ({{50, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_8_BIT** ({{63, 11}})

- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_8_BIT** ({{74, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_8_BIT** ({{90, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_8_BIT** ({{101, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_8_BIT** ({{117, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_9_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A_9_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T0A_9_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_9_BIT** ({{23, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_9_BIT** ({{39, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_9_BIT** ({{55, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_9_BIT** ({{68, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_9_BIT** ({{70, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D3_9_BIT** ({{72, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_9_BIT** ({{78, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_9_BIT** ({{91, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_9_BIT** ({{102, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_9_BIT** ({{118, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_10_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_10_BIT** ({{11, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_10_BIT** ({{27, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_10_BIT** ({{38, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_10_BIT** ({{54, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_10_BIT** ({{70, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_10_BIT** ({{83, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_10_BIT** ({{85, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **A_0_G_10_BIT** ({{87, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1_G_10_BIT** ({{103, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **T_0_G_10_BIT** ({{115, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_0_G_10_BIT** ({{123, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_DeltaAred_BIT** ({{7, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_exred_BIT** ({{12, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_eyred_BIT** ({{25, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_Deltai0red_BIT** ({{38, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_Omega0red_BIT** ({{55, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_lambda0red_BIT** ({{78, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_af0red_BIT** ({{101, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_af1red_BIT** ({{123, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **RS_IODNAV_LSBS** ({{15, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_CONSTELLATION_ID_BIT** ({{7, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_SERVICE_LEVEL_ID_BIT** ({{10, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_WN_BIT** ({{13, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_T0_BIT** ({{25, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_MASK_MSB_BIT** ({{34, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_MASK_BIT** ({{35, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_PCONST_BIT** ({{67, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_PSAT_BIT** ({{71, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_URA_BIT** ({{75, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_URE_BIT** ({{79, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_BNOM_BIT** ({{83, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_TVALIDITY_BIT** ({{87, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_CRC_BIT** ({{97, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **TIME_0_BIT** ({{7, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_0_BIT** ({{97, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_0_BIT** ({{109, 20}})
- const std::string **PEMFILE_DEFAULT** ("./OSNMA_PublicKey.pem")

- const std::string **CRTFILE_DEFAULT** ("./OSNMA_PublicKey.crt")
- const std::string **MERKLEFILE_DEFAULT** ("./OSNMA_MerkleTree.xml")
- const std::string **KROOTFILE_DEFAULT** ("./OSNMA_DSM_KROOT_NMAHeader.bin")
- const std::vector< std::pair< int32_t, int32_t > > **STRING_ID** ({{2, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **KX** ({{78, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **P1** ({{8, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_HR** ({{10, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_MIN** ({{15, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_SEC** ({{21, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **B_N** ({{6, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **P2** ({{9, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **T_B** ({{10, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **P3** ({{6, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **GAMMA_N** ({{7, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **P** ({{19, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **EPH_L_N** ({{21, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_N** ({{6, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_TAU_N** ({{28, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **E_N** ({{33, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **P4** ({{52, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **F_T** ({{53, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **N_T** ({{60, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **N** ({{71, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **M** ({{76, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **DAY_NUMBER_A** ({{6, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_C** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **N_4** ({{50, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_GPS** ({{55, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_L_N** ({{77, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **C_N** ({{6, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **M_N_A** ({{7, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **N_A** ({{9, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_N_A** ({{14, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **LAMBDA_N_A** ({{24, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_N_A** ({{45, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **EPSILON_N_A** ({{63, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_N_A** ({{6, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **T_LAMBDA_N_A** ({{22, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_N_A** ({{43, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_DOT_N_A** ({{65, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **H_N_A** ({{72, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **B1** ({{6, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **B2** ({{17, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **KP** ({{27, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PRN** ({{9, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_MSG_TYPE** ({{15, 6}})

- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOW** ({{21, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALERT_FLAG** ({{38, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN** ({{39, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_HEALTH** ({{52, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOP1** ({{55, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA** ({{66, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOE1** ({{71, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_A** ({{82, 26}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A_DOT** ({{108, 25}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_N0** ({{133, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_N0_DOT** ({{150, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_M0** ({{173, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_E_ECCENTRICITY** ({{206, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_OMEGA** ({{239, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_INTEGRITY_FLAG** ({{272, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_L2_PHASING_FLAG** ({{273, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOE2** ({{39, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_OMEGA0** ({{50, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_I0** ({{83, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_OMEGA_DOT** ({{116, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_I0_DOT** ({{133, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CIS** ({{148, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CIC** ({{164, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CRS** ({{180, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CRC** ({{204, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CUS** ({{228, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CUC** ({{249, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOP2** ({{39, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED0** ({{50, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED1** ({{55, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED2** ({{58, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOC** ({{61, 11}})
- const std::vector< std::pair< int, int > > **CNAV_AF0** ({{72, 26}})
- const std::vector< std::pair< int, int > > **CNAV_AF1** ({{98, 20}})
- const std::vector< std::pair< int, int > > **CNAV_AF2** ({{118, 10}})
- const std::vector< std::pair< int, int > > **CNAV_TGD** ({{128, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL1** ({{141, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL2** ({{154, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL5I** ({{167, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL5Q** ({{180, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA0** ({{193, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA1** ({{201, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA2** ({{209, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA3** ({{217, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA0** ({{225, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA1** ({{233, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA2** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA3** ({{249, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WNOP** ({{257, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_T_EOP** ({{128, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_X** ({{144, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_X_DOT** ({{165, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_Y** ({{180, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_Y_DOT** ({{201, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_UT1_GPS** ({{216, 31}})

- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_UT1_GPS_DOT** ({{247, 19}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A0** ({{128, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A1** ({{144, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A2** ({{157, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_TLS** ({{164, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOT** ({{172, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN_OT** ({{188, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN_LSF** ({{201, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DN** ({{214, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_TLSF** ({{218, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW** ({{31, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **INTEGRITY_STATUS_FLAG** ({{23, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ALERT_FLAG** ({{48, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ANTI_SPOOFING_FLAG** ({{49, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **SUBFRAME_ID** ({{50, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **GPS_WEEK** ({{61, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **CA_OR_P_ON_L2** ({{71, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_ACCURACY** ({{73, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_HEALTH** ({{77, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **L2_P_DATA_FLAG** ({{91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **T_GD** ({{197, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **IODC** ({{83, 2}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OC** ({{219, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F2** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F1** ({{249, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F0** ({{271, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **IODE_SF2** ({{61, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RS** ({{69, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_N** ({{91, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **M_0** ({{107, 8}, {121, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_UC** ({{151, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ECCENTRICITY** ({{167, 8}, {181, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_US** ({{211, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **SQRT_A** ({{227, 8}, {241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OE** ({{271, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FIT_INTERVAL_FLAG** ({{287, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **AODO** ({{288, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IC** ({{61, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_0** ({{77, 8}, {91, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IS** ({{121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **I_0** ({{137, 8}, {151, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RC** ({{181, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA** ({{197, 8}, {211, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT** ({{241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **IODE_SF3** ({{271, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **I_DOT** ({{279, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_DATA_ID** ({{61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_PAGE** ({{63, 6}})
- const std::vector< std::pair< int32_t, int32_t > > ALPHA_0 ({{69, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_1** ({{77, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_2** ({{91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_3** ({{99, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_0** ({{107, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_1** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_2** ({{129, 8}})

- const std::vector< std::pair< int32_t, int32_t > > **BETA_3** ({{137, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1** ({{151, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **A_0** ({{181, 24}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OT** ({{219, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_T** ({{227, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTAT_LS** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_LSF** ({{249, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DN** ({{257, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTAT_LSF** ({{271, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV1** ({{69, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV2** ({{73, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV3** ({{77, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV4** ({{81, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV5** ({{91, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV6** ({{95, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV7** ({{99, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV8** ({{103, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV9** ({{107, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV10** ({{111, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV11** ({{121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV12** ({{125, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV13** ({{129, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV14** ({{133, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV15** ({{137, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV16** ({{141, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV17** ({{151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV18** ({{155, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV19** ({{159, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV20** ({{163, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV21** ({{167, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV22** ({{171, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV23** ({{181, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV24** ({{185, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV25** ({{189, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV26** ({{193, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV27** ({{197, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV28** ({{201, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV29** ({{211, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV30** ({{215, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV31** ({{219, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV32** ({{223, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV25** ({{229, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV26** ({{241, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV27** ({{247, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV28** ({{253, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV29** ({{259, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV30** ({{271, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV31** ({{277, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV32** ({{283, 6}})
- const std::vector< std::pair< int32_t, int32_t > > T_OA ({{69, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A** ({{77, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV1** ({{91, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV2** ({{97, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV3** ({{103, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV4** ({{109, 6}})

- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV5** ({{121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV6** ({{127, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV7** ({{133, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV8** ({{139, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV9** ({{151, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV10** ({{157, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV11** ({{163, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV12** ({{169, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV13** ({{181, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV14** ({{187, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV15** ({{193, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV16** ({{199, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV17** ({{211, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV18** ({{217, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV19** ({{223, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV20** ({{229, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV21** ({{241, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV22** ({{247, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV23** ({{253, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV24** ({{259, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_ECC** ({{69, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_TOA** ({{91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_DELTAI** ({{99, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGADOT** ({{121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_SVHEALTH** ({{137, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_SQUAREA** ({{151, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGAZERO** ({{181, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGA** ({{211, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_MZERO** ({{241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_AF0** ({{271, 8}, {290, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_AF1** ({{279, 11}})
- int32_t gps_ref_week_from_config (const std::string &observation_date, bool pre_2009_file)

  Derives the reference GPS week number used to resolve the GPS mod-1024 week-number rollover when post-processing recorded signal files.
- constexpr uint32_t qzss_l1cb_prn_to_nominal_prn (uint32_t prn)

  Maps QZSS L1 C/B PRNs to the PRN carrying the satellite's nominal PNT signals. L1 C/B is broadcast with a dedicated PRN (203-206), but the transmitting satellite is identified by the PRN of its nominal signals (RINEX 4.00, Table 6: QZSS PRN to RINEX Satellite Identifier): 203->196 (J04), 204->197 (J05), 205->200 (J08), 206->201 (J09). Returns the input PRN unchanged if it is not an L1 C/B alias; in particular, PRN 202 (J10) broadcasts L1 C/B under its own PRN.

Variables

- constexpr double BEIDOU_B1C_FREQ_HZ = FREQ1

  §4.2.2, Table 4-1 [Hz]
- constexpr double BEIDOU_B1C_CODE_RATE_CPS = 1.023e6

  §5.2.1 [chips/s]
- constexpr double BEIDOU_B1C_CODE_LENGTH_CHIPS = 10230.0

  §5.2.1 [chips]
- constexpr double BEIDOU_B1C_CODE_PERIOD_S = 0.01

  §5.2.1, 10 ms [s]
- constexpr uint32_t BEIDOU_B1C_CODE_PERIOD_MS = 10

  §5.2.1 [ms]
- constexpr uint32_t BEIDOU_B1C_OPT_ACQ_FS_SPS = 4000000

4 samples/chip at 1.023 Mcps

- constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH = 1800

  §5.2.2 [chips]
- constexpr int32_t BEIDOU_B1C_TELEMETRY_RATE_SPS = 100

  §6.2 [symbols/s]
- constexpr int32_t BEIDOU_B1C_SYMBOLS_PER_BIT = 1

  100 sps, 10 ms/code period
- constexpr int32_t BEIDOU_B1C_FRAME_LENGTH_SYMBOLS = 1800

  §6.2 [symbols]
- constexpr int32_t BEIDOU_B1C_FRAME_PERIOD_S = 18

  §6.2 [s]
- constexpr int32_t BEIDOU_B1C_NUMBER_OF_PRNS = 63

  §5.2.1
- constexpr double BEIDOU_B1I_FREQ_HZ = FREQ1_BDS

  B1I [Hz].
- constexpr double BEIDOU_B1I_CODE_RATE_CPS = 2.046e6

  Beidou B1I code rate [chips/s].
- constexpr double BEIDOU_B1I_CODE_LENGTH_CHIPS = 2046.0

  Beidou B1I code length [chips].
- constexpr double BEIDOU_B1I_CODE_PERIOD_S = 0.001

  Beidou B1I code period [seconds].
- constexpr double BEIDOU_B1I_PREAMBLE_DURATION_S = 0.220
- constexpr double BEIDOU_B1I_OPT_ACQ_FS_SPS = 4e6

  ∼2 samples/chip (same ratio as GPS L1 at 2 MHz)
- constexpr uint32_t BEIDOU_B1I_CODE_PERIOD_MS = 1

  Beidou B1I code period [ms].
- constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_BITS = 11
- constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS = 220
- constexpr int32_t BEIDOU_B1I_SECONDARY_CODE_LENGTH = 20
- constexpr int32_t BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22
- constexpr int32_t BEIDOU_B1I_PREAMBLE_DURATION_MS = 220
- constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND = 50
- constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2
- constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_↩ cast<uint32_t>(BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT) ∗ BEIDOU_B1I_CODE_PERIOD_MS)
- constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B1I_TELEMETRY↩ _RATE_BITS_SECOND ∗ BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT
- constexpr char BEIDOU_B1I_SECONDARY_CODE_STR [21] = "00000100110101001110"
- constexpr char BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR [23] = "11111100000011000011100"
- constexpr char BEIDOU_B1I_D2_SECONDARY_CODE_STR [3] = "00"
- constexpr double BEIDOU_B2A_FREQ_HZ = FREQ5

  BeiDou B2a [Hz].
- constexpr double BEIDOU_B2A_CODE_RATE_CPS = 10.23e6

  B2a primary code rate [chips/s].
- constexpr double BEIDOU_B2A_CODE_LENGTH_CHIPS = 10230.0

  B2a primary code length [chips].
- constexpr double BEIDOU_B2A_CODE_PERIOD_S = 0.001

  B2a primary code period [seconds].
- constexpr double BEIDOU_B2A_OPT_ACQ_FS_SPS = 20e6
- constexpr uint32_t BEIDOU_B2A_CODE_PERIOD_MS = 1
- constexpr uint32_t BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS = 24

- constexpr int32_t BEIDOU_B2A_SYMBOLS_PER_BIT = 1
- constexpr int32_t BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND = 1000
- constexpr int32_t BEIDOU_B2A_SECONDARY_CODE_LENGTH = 5
- constexpr char BEIDOU_B2A_SECONDARY_CODE_STR [6] = "00010"

  ICD data secondary code, MSB first.
- constexpr char BEIDOU_B2A_PREAMBLE_SYMBOLS_STR [25] = "11100010010011010111101000"

  0xE24DE8
- constexpr double BEIDOU_B3I_FREQ_HZ = FREQ3_BDS

  BeiDou B3I [Hz].
- constexpr double BEIDOU_B3I_CODE_RATE_CPS = 10.23e6

  BeiDou B3I code rate [chips/s].
- constexpr double BEIDOU_B3I_CODE_LENGTH_CHIPS = 10230.0

  BeiDou B3I code length [chips].
- constexpr double BEIDOU_B3I_CODE_PERIOD_S = 0.001

  BeiDou B3I code period [seconds].
- constexpr double BEIDOU_B3I_PREAMBLE_DURATION_S = 0.220
- constexpr double BEIDOU_B3I_OPT_ACQ_FS_SPS = 10e6
- constexpr uint32_t BEIDOU_B3I_CODE_PERIOD_MS = 1

  BeiDou B3I code period [ms].
- constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_BITS = 11
- constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS = 220
- constexpr int32_t BEIDOU_B3I_SECONDARY_CODE_LENGTH = 20
- constexpr int32_t BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22
- constexpr int32_t BEIDOU_B3I_PREAMBLE_DURATION_MS = 220
- constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND = 50

  D1 NAV message bit rate [bits/s].
- constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2
- constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_↩ cast<uint32_t>(BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT) * BEIDOU_B3I_CODE_PERIOD_MS)
- constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B3I_TELEMETRY_RATE_BITS_SECO * BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT
- constexpr char BEIDOU_B3I_SECONDARY_CODE_STR [21] = "00000100110101001110"
- constexpr char BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR [23] = "11111100000011000011100"
- constexpr char BEIDOU_B3I_D2_SECONDARY_CODE_STR [3] = "00"
- constexpr int32_t BEIDOU_CNAV1_FRAME_SYMBOLS = 1800
- constexpr int32_t BEIDOU_CNAV1_FRAME_PERIOD_S = 18
- constexpr int32_t BEIDOU_CNAV1_SYMBOL_RATE_SPS = 100
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME1_SYMBOLS = 72
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME2_SYMBOLS = 1200
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME3_SYMBOLS = 528
- constexpr int32_t BEIDOU_CNAV1_SOH_LSB_S = 18
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVED_SYMBOLS = 1728
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_ROWS = 36
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_COLS = 48
- constexpr int32_t BEIDOU_CNAV1_SF2_DATA_BITS = 600
- constexpr int32_t BEIDOU_CNAV1_SF3_DATA_BITS = 264
- constexpr int32_t BEIDOU_CNAV1_CRC_BITS = 24
- constexpr int32_t BEIDOU_CNAV1_PAGE_IONO_UTC = 1
- constexpr int32_t BDS_EPH_SOURCE_CNAV1 = 7

  RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV1 (DNAV uses 0–6).
- constexpr double BEIDOU_CNAV1_A_REF_MEO = 27906100.0
- constexpr double BEIDOU_CNAV1_A_REF_IGSO = 42162200.0

- constexpr double BEIDOU_CNAV1_TOE_TOC_LSB = 300.0
- constexpr double BEIDOU_CNAV1_DELTA_A_LSB = TWO_N9
- constexpr double BEIDOU_CNAV1_A_DOT_LSB = TWO_N21

  2^-21 m/s, Table 7-8
- constexpr double BEIDOU_CNAV1_DELTA_N0_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI
- constexpr double BEIDOU_CNAV1_M0_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_E_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_OMEGA_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_I0_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_OMEGADOT_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_IDOT_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_CIS_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CIC_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CRS_LSB = TWO_N8
- constexpr double BEIDOU_CNAV1_CRC_LSB = TWO_N8
- constexpr double BEIDOU_CNAV1_CUS_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CUC_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_AF0_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_AF1_LSB = TWO_N50
- constexpr double BEIDOU_CNAV1_AF2_LSB = TWO_N66
- constexpr double BEIDOU_CNAV1_TGD_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_ISC_LSB = TWO_N34
- constexpr int32_t BEIDOU_CNAV1_LDPC_N200 = 200
- constexpr int32_t BEIDOU_CNAV1_LDPC_K200 = 100
- constexpr int32_t BEIDOU_CNAV1_LDPC_M200 = 100
- constexpr int32_t BEIDOU_CNAV1_LDPC_DC200 = 4
- constexpr int32_t BEIDOU_CNAV1_LDPC_N88 = 88
- constexpr int32_t BEIDOU_CNAV1_LDPC_K88 = 44
- constexpr int32_t BEIDOU_CNAV1_LDPC_M88 = 44
- constexpr int32_t BEIDOU_CNAV1_LDPC_DC88 = 4
- constexpr int32_t BEIDOU_CNAV1_LDPC_NM = BEIDOU_LDPC_NM
- constexpr int32_t BEIDOU_CNAV1_LDPC_MAX_ITER = BEIDOU_LDPC_MAX_ITER
- constexpr uint16_t BEIDOU_CNAV1_H100_200_INDEX [400]
- constexpr uint8_t BEIDOU_CNAV1_H100_200_ELEMENT [400]
- constexpr uint16_t BEIDOU_CNAV1_H44_88_INDEX [176]
- constexpr uint8_t BEIDOU_CNAV1_H44_88_ELEMENT [176]
- constexpr int32_t BEIDOU_CNAV2_FRAME_SYMBOLS = 600
- constexpr int32_t BEIDOU_CNAV2_FRAME_PERIOD_S = 3
- constexpr int32_t BEIDOU_CNAV2_SYMBOL_RATE_SPS = 200
- constexpr int32_t BEIDOU_CNAV2_MS_PER_SYMBOL = 5
- constexpr int32_t BEIDOU_CNAV2_FRAME_MS = 3000
- constexpr int32_t BEIDOU_CNAV2_PREAMBLE_SYMBOLS = 24
- constexpr int32_t BEIDOU_CNAV2_PREAMBLE_MS = 120

  *24 symbols * 5 ms secondary code*
- constexpr int32_t BEIDOU_CNAV2_LDPC_SYMBOLS = 576

  encoded payload: 96 GF(64) symbols, 6 bits each
- constexpr int32_t BEIDOU_CNAV2_INFO_BITS = 288
- constexpr int32_t BEIDOU_CNAV2_CRC_BITS = 24
- constexpr int32_t BEIDOU_CNAV2_DATA_BITS = BEIDOU_CNAV2_INFO_BITS - BEIDOU_CNAV2_CRC↩ _BITS
- constexpr int32_t BEIDOU_CNAV2_MES_TYPE_BITS = 6
- constexpr int32_t BEIDOU_CNAV2_SOW_LSB_S = 3
- constexpr double BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD = 5.2

on the normalized soft preamble statistic, about \|N(0,1)\| on noise

- constexpr int32_t BEIDOU_CNAV2_MSG_EPH1 = 10
- constexpr int32_t BEIDOU_CNAV2_MSG_EPH2 = 11
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_IONO = 30
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_ALM = 31
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_EOP = 32
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_UTC = 33
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_DC = 34
- constexpr int32_t BEIDOU_CNAV2_MSG_ALM = 40
- constexpr int32_t BEIDOU_CNAV2_MT10_IODE_BIT = 53
- constexpr int32_t BEIDOU_CNAV2_IODE_BITS = 8
- constexpr int32_t BDS_EPH_SOURCE_CNAV2 = 9

RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV2.

- constexpr int32_t BEIDOU_CNAV2_LDPC_N = 96
- constexpr int32_t BEIDOU_CNAV2_LDPC_K = 48
- constexpr int32_t BEIDOU_CNAV2_LDPC_M = 48
- constexpr int32_t BEIDOU_CNAV2_LDPC_DC = 4
- constexpr uint16_t BEIDOU_CNAV2_H48_96_INDEX [192]
- constexpr uint8_t BEIDOU_CNAV2_H48_96_ELEMENT [192]
- constexpr double D1_TOC_LSB = TWO_P3
- constexpr double D1_TGD1_LSB = 0.1e-9
- constexpr double D1_TGD2_LSB = 0.1e-9
- constexpr double D1_ALPHA0_LSB = TWO_N30
- constexpr double D1_ALPHA1_LSB = TWO_N27
- constexpr double D1_ALPHA2_LSB = TWO_N24
- constexpr double D1_ALPHA3_LSB = TWO_N24
- constexpr double D1_BETA0_LSB = TWO_P11
- constexpr double D1_BETA1_LSB = TWO_P14
- constexpr double D1_BETA2_LSB = TWO_P16
- constexpr double D1_BETA3_LSB = TWO_P16
- constexpr double D1_A2_LSB = TWO_N66
- constexpr double D1_A0_LSB = TWO_N33
- constexpr double D1_A1_LSB = TWO_N50
- constexpr double D1_DELTA_N_LSB = PI_TWO_N43
- constexpr double D1_CUC_LSB = TWO_N31
- constexpr double D1_M0_LSB = PI_TWO_N31
- constexpr double D1_E_LSB = TWO_N33
- constexpr double D1_CUS_LSB = TWO_N31
- constexpr double D1_CRC_LSB = TWO_N6
- constexpr double D1_CRS_LSB = TWO_N6
- constexpr double D1_SQRT_A_LSB = TWO_N19
- constexpr double D1_TOE_LSB = TWO_P3
- constexpr double D1_I0_LSB = PI_TWO_N31
- constexpr double D1_CIC_LSB = TWO_N31
- constexpr double D1_OMEGA_DOT_LSB = PI_TWO_N43
- constexpr double D1_CIS_LSB = TWO_N31
- constexpr double D1_IDOT_LSB = PI_TWO_N43
- constexpr double D1_OMEGA0_LSB = PI_TWO_N31
- constexpr double D1_OMEGA_LSB = PI_TWO_N31
- constexpr double D1_SQRT_A_ALMANAC_LSB = TWO_N11
- constexpr double D1_A1_ALMANAC_LSB = TWO_N38
- constexpr double D1_A0_ALMANAC_LSB = TWO_N20
- constexpr double D1_OMEGA0_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_E_ALMANAC_LSB = TWO_N21

- constexpr double D1_DELTA_I_LSB = PI_TWO_N19
- constexpr double D1_TOA_LSB = TWO_P12
- constexpr double D1_OMEGA_DOT_ALMANAC_LSB = PI_TWO_N38
- constexpr double D1_OMEGA_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_M0_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_A0GPS_LSB = 0.1e-9
- constexpr double D1_A1GPS_LSB = 0.1e-9
- constexpr double D1_A0GAL_LSB = 0.1e-9
- constexpr double D1_A1GAL_LSB = 0.1e-9
- constexpr double D1_A0GLO_LSB = 0.1e-9
- constexpr double D1_A1GLO_LSB = 0.1e-9
- constexpr double D1_A0UTC_LSB = TWO_N30
- constexpr double D1_A1UTC_LSB = TWO_N50
- constexpr double D2_DELTA_T_LSB = 0.1
- constexpr double D2_ION_DELAY_LSB = 0.125
- constexpr std::size_t BEIDOU_DNAV_BCH_CODE_BITS = 15
- constexpr std::size_t BEIDOU_DNAV_BCH_DATA_BITS = 11
- constexpr std::size_t BEIDOU_DNAV_BCH_PARITY_BITS = 4
- constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_BITS = 11
- constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS = 11
- constexpr int32_t BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS = 300
- constexpr int32_t BEIDOU_DNAV_SUBFRAME_DATA_BITS = 300

  Number of bits per subframe in the NAV message [bits].
- constexpr double BEIDOU_DNAV_SECONDS_PER_WEEK = 604800.0
- constexpr double BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS = 6.0
- constexpr double BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS = 3.0
- constexpr int32_t BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET = 14
- constexpr int32_t BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET = 1356
- constexpr uint32_t BEIDOU_DNAV_SUBFRAME_SYMBOLS = 300
- constexpr uint32_t BEIDOU_DNAV_WORDS_SUBFRAME = 10
- constexpr uint32_t BEIDOU_DNAV_WORD_LENGTH_BITS = 30
- constexpr char BEIDOU_DNAV_PREAMBLE [12] = "11100010010"
- constexpr int32_t D2_UDREI1_LOGICAL_FIRST_BIT = 172
- constexpr int32_t D2_UDREI19_LOGICAL_FIRST_BIT = 220
- constexpr int32_t D2_ION_LOGICAL_FIRST_BIT = 51
- const std::string TEXT_RESET = "\033[0m"
- const std::string TEXT_BLACK = "\033[30m"
- const std::string TEXT_RED = "\033[31m"
- const std::string TEXT_GREEN = "\033[32m"
- const std::string TEXT_YELLOW = "\033[33m"
- const std::string TEXT_BLUE = "\033[34m"
- const std::string TEXT_MAGENTA = "\033[35m"
- const std::string TEXT_CYAN = "\033[36m"
- const std::string TEXT_WHITE = "\033[37m"
- const std::string TEXT_BOLD_BLACK = "\033[1m\033[30m"
- const std::string TEXT_BOLD_RED = "\033[1m\033[31m"
- const std::string TEXT_BOLD_GREEN = "\033[1m\033[32m"
- const std::string TEXT_BOLD_YELLOW = "\033[1m\033[33m"
- const std::string TEXT_BOLD_BLUE = "\033[1m\033[34m"
- const std::string TEXT_BOLD_MAGENTA = "\033[1m\033[35m"
- const std::string TEXT_BOLD_CYAN = "\033[1m\033[36m"
- const std::string TEXT_BOLD_WHITE = "\033[1m\033[37m"
- constexpr size_t HAS_MSG_NSYS_LENGTH = 4
- constexpr size_t HAS_MSG_ID_MASK_LENGTH = 4

- constexpr size_t HAS_MSG_SATELLITE_MASK_LENGTH = 40
- constexpr size_t HAS_MSG_SIGNAL_MASK_LENGTH = 16
- constexpr size_t HAS_MSG_NAV_MESSAGE_LENGTH = 3
- constexpr size_t HAS_MSG_MASK_RESERVED_LENGTH = 6
- constexpr size_t HAS_MSG_VALIDITY_INDEX_LENGTH = 4
- constexpr size_t HAS_MSG_IOD_GPS_LENGTH = 8
- constexpr size_t HAS_MSG_IOD_GAL_LENGTH = 10
- constexpr size_t HAS_MSG_DELTA_RADIAL_LENGTH = 13
- constexpr size_t HAS_MSG_DELTA_IN_TRACK_LENGTH = 12
- constexpr size_t HAS_MSG_DELTA_CROSS_TRACK_LENGTH = 12
- constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH = 2
- constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH = 13
- constexpr size_t HAS_MSG_NSYS_SUB_LENGTH = 4
- constexpr size_t HAS_MSG_ID_CLOCK_SUBSET_LENGTH = 4
- constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH = 2
- constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH = 13
- constexpr size_t HAS_MSG_CODE_BIAS_LENGTH = 11
- constexpr size_t HAS_MSG_PHASE_BIAS_LENGTH = 11
- constexpr size_t HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH = 2
- constexpr uint64_t MAX_SECONDS_REMEMBERING_MID = 150
- constexpr uint64_t MAX_SECONDS_REMEMBERING_MASK_IOD = 1800
- constexpr uint32_t GALILEO_HAS_SECONDS_PER_HOUR = 3600U
- constexpr uint32_t GALILEO_HAS_SECONDS_PER_WEEK = 604800U
- constexpr uint32_t GALILEO_HAS_HALF_WEEK_SECONDS = GALILEO_HAS_SECONDS_PER_WEEK / 2U
- constexpr uint32_t GALILEO_HAS_INVALID_WEEK = std::numeric_limits<uint32_t>::max()
- constexpr uint32_t GALILEO_HAS_INVALID_TOW = std::numeric_limits<uint32_t>::max()
- constexpr int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000
- constexpr int32_t GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS = 1000
- constexpr int32_t GALILEO_CNAV_PAGE_MS = 1
- constexpr int32_t GALILEO_CNAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_CNAV_INTERLEAVER_COLS = 123
- constexpr int32_t GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND = 1000
- constexpr int32_t GALILEO_CNAV_HAS_PAGE_DATA_BITS = 448
- constexpr int32_t GALILEO_CNAV_PAGE_RESERVED_BITS = 14
- constexpr int32_t GALILEO_CNAV_BYTES_FOR_CRC = 58
- constexpr int32_t GALILEO_CNAV_CRC_LENGTH = 24
- constexpr int32_t GALILEO_CNAV_MESSAGE_BITS_PER_PAGE = 424
- constexpr int32_t GALILEO_CNAV_PAGE_HEADER_BITS = 24
- constexpr int32_t GALILEO_CNAV_PREAMBLE_LENGTH_BITS = 16
- constexpr int32_t GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK = 255
- constexpr int32_t GALILEO_CNAV_MT1_HEADER_BITS = 32
- constexpr int32_t GALILEO_CNAV_OCTETS_IN_SUBPAGE = 53
- constexpr int32_t GALILEO_CNAV_INFORMATION_VECTOR_LENGTH = 32
- constexpr int32_t GALILEO_CNAV_BITS_FOR_CRC = GALILEO_CNAV_HAS_PAGE_DATA_BITS + GALILEO_CNAV_PAGE_RESERVED_BITS
- constexpr int32_t HAS_MSG_NUMBER_MASK_IDS = 32
- constexpr int32_t HAS_MSG_NUMBER_GNSS_IDS = 16
- constexpr int32_t HAS_MSG_NUMBER_MESSAGE_IDS = 32
- constexpr int32_t HAS_MSG_NUMBER_SATELLITE_IDS = 40
- constexpr int32_t HAS_MSG_NUMBER_SIGNAL_MASKS = 16
- constexpr float HAS_MSG_DELTA_RADIAL_SCALE_FACTOR = 0.0025
- constexpr float HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR = 0.008
- constexpr float HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR = 0.008
- constexpr float HAS_MSG_DELTA_CLOCK_SCALE_FACTOR = 0.0025

- constexpr float HAS_MSG_CODE_BIAS_SCALE_FACTOR = 0.02
- constexpr float HAS_MSG_PHASE_BIAS_SCALE_FACTOR = 0.01
- constexpr uint16_t HAS_MSG_NUMBER_MAX_TOH = 3599
- constexpr uint8_t HAS_MSG_GPS_SYSTEM = 0
- constexpr uint8_t HAS_MSG_GALILEO_SYSTEM = 2
- constexpr uint8_t HAS_MSG_WRONG_SYSTEM = 255
- constexpr char GALILEO_CNAV_PREAMBLE [17] = "1011011101110000"
- const std::unordered_map< uint8_t, std::unordered_map< uint8_t, std::string > > HAS_SIGNAL_INDEX_TABLE
- const std::unordered_map< uint8_t, uint16_t > HAS_VALIDITY_INTERVALS
- constexpr double GALILEO_E1_FREQ_HZ = FREQ1

  Galileo E1 carrier frequency [Hz].
- constexpr double GALILEO_E1_CODE_CHIP_RATE_CPS = 1.023e6

  Galileo E1 code rate [chips/s].
- constexpr double GALILEO_E1_CODE_PERIOD_S = 0.004

  Galileo E1 code period [s].
- constexpr double GALILEO_E1_SUB_CARRIER_A_RATE_HZ = 1.023e6

  Galileo E1 sub-carrier 'a' rate [Hz].
- constexpr double GALILEO_E1_SUB_CARRIER_B_RATE_HZ = 6.138e6

  Galileo E1 sub-carrier 'b' rate [Hz].
- constexpr double GALILEO_E1_B_CODE_LENGTH_CHIPS = 4092.0

  Galileo E1-B code length [chips].
- constexpr double GALILEO_E1_B_SYMBOL_RATE_BPS = 250.0

  Galileo E1-B symbol rate [bits/second].
- constexpr uint32_t GALILEO_E1_CODE_PERIOD_MS = 4

  Galileo E1 code period [ms].
- constexpr int32_t GALILEO_E1_B_SAMPLES_PER_SYMBOL = 1

  (Galileo_E1_CODE_CHIP_RATE_HZ / Galileo_E1_B_CODE_LENGTH_CHIPS) / Galileo_E1_B_SYMBOL_RATE↩ _BPS
- constexpr int32_t GALILEO_E1_C_SECONDARY_CODE_LENGTH = 25

  Galileo E1-C secondary code length [chips].
- constexpr int32_t GALILEO_E1_NUMBER_OF_CODES = 50
- constexpr uint32_t GALILEO_E1_OPT_ACQ_FS_SPS = 2000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GALILEO_E1_HISTORY_DEEP = 100

  Observable history length for interpotalion.
- constexpr char GALILEO_E1_C_SECONDARY_CODE [26] = "00111000000010101101100 10"
- constexpr size_t GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH = 1023
- constexpr char GALILEO_E1_B_PRIMARY_CODE [GALILEO_E1_NUMBER_OF_CODES][1024]
- constexpr size_t GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH = 1023
- constexpr char GALILEO_E1_C_PRIMARY_CODE [GALILEO_E1_NUMBER_OF_CODES][1024]
- constexpr double GALILEO_E5A_FREQ_HZ = FREQ5

  Galileo E5a carrier frequency [Hz].
- constexpr double GALILEO_E5A_CODE_CHIP_RATE_CPS = 1.023e7

  Galileo E5a code rate [chips/s].
- constexpr double GALILEO_E5A_I_TIERED_CODE_PERIOD_S = 0.020

  Galileo E5a-I tiered code period [s].
- constexpr double GALILEO_E5A_Q_TIERED_CODE_PERIOD_S = 0.100

  Galileo E5a-Q tiered code period [s].
- constexpr double GALILEO_E5A_CODE_PERIOD_S = 0.001

  Galileo E5a primary code period [s].
- constexpr int32_t GALILEO_E5A_CODE_LENGTH_CHIPS = 10230

  Galileo E5a primary code length [chips].

- constexpr int32_t GALILEO_E5A_I_SECONDARY_CODE_LENGTH = 20

  Galileo E5a-I secondary code length [chips].
- constexpr int32_t GALILEO_E5A_Q_SECONDARY_CODE_LENGTH = 100

  Galileo E5a-Q secondary code length [chips].
- constexpr int32_t GALILEO_E5A_CODE_PERIOD_MS = 1

  Galileo E5a primary code period [ms].
- constexpr int32_t GALILEO_E5A_SYMBOL_RATE_BPS = 50

  Galileo E5a symbol rate [bits/second].
- constexpr int32_t GALILEO_E5A_NUMBER_OF_CODES = 50
- constexpr int32_t GALILEO_E5A_HISTORY_DEEP = 20
- constexpr int32_t GALILEO_E5A_CRC_ERROR_LIMIT = 6
- constexpr uint32_t GALILEO_E5A_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GALILEO_FNAV_PREAMBLE_LENGTH_BITS = 12
- constexpr int32_t GALILEO_FNAV_CODES_PER_SYMBOL = 20
- constexpr int32_t GALILEO_FNAV_CODES_PER_PREAMBLE = 240
- constexpr int32_t GALILEO_FNAV_SYMBOLS_PER_PAGE = 500
- constexpr int32_t GALILEO_FNAV_SECONDS_PER_PAGE = 10
- constexpr int32_t GALILEO_FNAV_CODES_PER_PAGE = 10000
- constexpr int32_t GALILEO_FNAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_FNAV_INTERLEAVER_COLS = 61
- constexpr int32_t GALILEO_FNAV_PAGE_TYPE_BITS = 6
- constexpr int32_t GALILEO_FNAV_DATA_FRAME_BITS = 214
- constexpr int32_t GALILEO_FNAV_DATA_FRAME_BYTES = 27
- constexpr char GALILEO_FNAV_PREAMBLE [13] = "101101110000"
- constexpr size_t GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5A_I_PRIMARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5A_Q_PRIMARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][2559]
- constexpr char GALILEO_E5A_I_SECONDARY_CODE [ ] = "10000100001011101001"
- constexpr size_t GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH = 100
- constexpr char GALILEO_E5A_Q_SECONDARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][101]
- constexpr double GALILEO_E5B_FREQ_HZ = FREQ7

  Galileo E5b carrier frequency [Hz].
- constexpr double GALILEO_E5B_CODE_CHIP_RATE_CPS = 1.023e7

  Galileo E5b code rate [chips/s].
- constexpr double GALILEO_E5B_I_TIERED_CODE_PERIOD_S = 0.004

  Galileo E5b-I tiered code period [s].
- constexpr double GALILEO_E5B_Q_TIERED_CODE_PERIOD_S = 0.100

  Galileo E5b-Q tiered code period [s].
- constexpr double GALILEO_E5B_CODE_PERIOD_S = 0.001

  Galileo E5b primary code period [s].
- constexpr int32_t GALILEO_E5B_CODE_PERIOD_MS = 1

  Galileo E5b primary code period [ms].
- constexpr int32_t GALILEO_E5B_CODE_LENGTH_CHIPS = 10230

  Galileo E5b primary code length [chips].
- constexpr int32_t GALILEO_E5B_I_SECONDARY_CODE_LENGTH = 4

  Galileo E5b-I secondary code length [chips].
- constexpr int32_t GALILEO_E5B_Q_SECONDARY_CODE_LENGTH = 100

  Galileo E5b-Q secondary code length [chips].
- constexpr int32_t GALILEO_E5B_SYMBOL_RATE_BPS = 250

  Galileo E5b symbol rate [bits/second].

- constexpr int32_t GALILEO_E5B_NUMBER_OF_CODES = 50
- constexpr int32_t GALILEO_E5B_HISTORY_DEEP = 100
- constexpr uint32_t GALILEO_E5B_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr char GALILEO_E5B_I_SECONDARY_CODE [5] = "1110"
- constexpr size_t GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5B_I_PRIMARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5B_Q_PRIMARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH = 100
- constexpr char GALILEO_E5B_Q_SECONDARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][101]
- constexpr double GALILEO_E6_FREQ_HZ = FREQ6

  Galileo E6 carrier frequency [Hz].
- constexpr double GALILEO_E6_B_CODE_CHIP_RATE_CPS = 5.115e6

  Galileo E6 B code rate [chips/s].
- constexpr double GALILEO_E6_C_CODE_CHIP_RATE_CPS = 5.115e6

  Galileo E6 C code rate [chips/s].
- constexpr double GALILEO_E6_CODE_PERIOD_S = 0.001

  Galileo E6 code period [s].
- constexpr double GALILEO_E6_B_CODE_LENGTH_CHIPS = 5115.0

  Galileo E6 B code length [chips].
- constexpr double GALILEO_E6_C_CODE_LENGTH_CHIPS = 5115.0

  Galileo E6 C code length [chips].
- constexpr double GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS = 100.0

  Galileo E6 C secondary code length [chips].
- constexpr uint32_t GALILEO_E6_CODE_PERIOD_MS = 1

  Galileo E& B/C code period [ms].
- constexpr int32_t GALILEO_E6_NUMBER_OF_CODES = 50
- constexpr uint32_t GALILEO_E6_OPT_ACQ_FS_SPS = 10000000
- constexpr size_t GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH = 1279
- constexpr char GALILEO_E6_B_PRIMARY_CODE [GALILEO_E6_NUMBER_OF_CODES][1280]
- constexpr size_t GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH = 1279
- constexpr char GALILEO_E6_C_PRIMARY_CODE [GALILEO_E6_NUMBER_OF_CODES][1280]
- constexpr size_t GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH = 25
- constexpr char GALILEO_E6_C_SECONDARY_CODE [GALILEO_E6_NUMBER_OF_CODES][26]
- constexpr int32_t FNAV_T0C_1_LSB = 60
- constexpr double FNAV_AF0_1_LSB = TWO_N34
- constexpr double FNAV_AF1_1_LSB = TWO_N46
- constexpr double FNAV_AF2_1_LSB = TWO_N59
- constexpr double FNAV_AI0_1_LSB = TWO_N2
- constexpr double FNAV_AI1_1_LSB = TWO_N8
- constexpr double FNAV_AI2_1_LSB = TWO_N15
- constexpr double FNAV_BGD_1_LSB = TWO_N32
- constexpr double FNAV_M0_2_LSB = PI_TWO_N31
- constexpr double FNAV_OMEGADOT_2_LSB = PI_TWO_N43
- constexpr double FNAV_E_2_LSB = TWO_N33
- constexpr double FNAV_A12_2_LSB = TWO_N19
- constexpr double FNAV_OMEGA0_2_LSB = PI_TWO_N31
- constexpr double FNAV_IDOT_2_LSB = PI_TWO_N43
- constexpr double FNAV_I0_3_LSB = PI_TWO_N31
- constexpr double FNAV_W_3_LSB = PI_TWO_N31
- constexpr double FNAV_DELTAN_3_LSB = PI_TWO_N43
- constexpr double FNAV_CUC_3_LSB = TWO_N29

- constexpr double FNAV_CUS_3_LSB = TWO_N29
- constexpr double FNAV_CRC_3_LSB = TWO_N5
- constexpr double FNAV_CRS_3_LSB = TWO_N5
- constexpr int32_t FNAV_T0E_3_LSB = 60
- constexpr double FNAV_CIC_4_LSB = TWO_N29
- constexpr double FNAV_CIS_4_LSB = TWO_N29
- constexpr double FNAV_A0_4_LSB = TWO_N30
- constexpr double FNAV_A1_4_LSB = TWO_N50
- constexpr int32_t FNAV_T0T_4_LSB = 3600
- constexpr int32_t FNAV_T0G_4_LSB = 3600
- constexpr double FNAV_A0G_4_LSB = TWO_N35
- constexpr double FNAV_A1G_4_LSB = TWO_N51
- constexpr int32_t FNAV_T0A_5_LSB = 600
- constexpr double FNAV_DELTAA12_5_LSB = TWO_N9
- constexpr double FNAV_E_5_LSB = TWO_N16
- constexpr double FNAV_W_5_LSB = TWO_N15
- constexpr double FNAV_DELTAI_5_LSB = TWO_N14
- constexpr double FNAV_OMEGA0_5_LSB = TWO_N15
- constexpr double FNAV_OMEGADOT_5_LSB = TWO_N33
- constexpr double FNAV_M0_5_LSB = TWO_N15
- constexpr double FNAV_AF0_5_LSB = TWO_N19
- constexpr double FNAV_AF1_5_LSB = TWO_N38
- constexpr double GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS = 2.04

  Page Duration + (Galileo I/NAV Preamble bits)∗(Galileo E5b-I tiered Code Period(seconds)).
- constexpr uint32_t GALILEO_INAV_PAGE_SYMBOLS = 500

  The complete Galileo INAV page length.
- constexpr int32_t GALILEO_INAV_PREAMBLE_LENGTH_BITS = 10
- constexpr int32_t GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS = 250
- constexpr int32_t GALILEO_INAV_PAGE_PART_SYMBOLS = 250

  Each Galileo INAV pages are composed of two parts (even and odd) each of 250 symbols, including preamble. See Galileo ICD 4.3.2.
- constexpr int32_t GALILEO_INAV_PAGE_PART_SECONDS = 1
- constexpr int32_t GALILEO_INAV_PAGE_PART_MS = 1000
- constexpr int32_t GALILEO_INAV_PAGE_SECONDS = 2
- constexpr int32_t GALILEO_INAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_INAV_INTERLEAVER_COLS = 30
- constexpr int32_t GALILEO_TELEMETRY_RATE_BITS_SECOND = 250
- constexpr int32_t GALILEO_PAGE_TYPE_BITS = 6
- constexpr int32_t GALILEO_DATA_JK_BITS = 128
- constexpr int32_t GALILEO_DATA_FRAME_BITS = 196
- constexpr int32_t GALILEO_DATA_FRAME_BYTES = 25
- constexpr size_t GALILEO_INAV_EVEN_PAGE_TYPE_BIT = 1
- constexpr size_t GALILEO_INAV_ODD_PAGE_TYPE_BIT = 115
- constexpr int32_t GALILEO_INAV_DUMMY_WORD_TYPE = 63
- constexpr char GALILEO_INAV_PREAMBLE [11] = "0101100000"
- constexpr int32_t T0E_1_LSB = 60
- constexpr double M0_1_LSB = PI_TWO_N31
- constexpr double E_1_LSB = TWO_N33
- constexpr double A_1_LSB_GAL = TWO_N19
- constexpr double OMEGA_0_2_LSB = PI_TWO_N31
- constexpr double I_0_2_LSB = PI_TWO_N31
- constexpr double OMEGA_2_LSB = PI_TWO_N31
- constexpr double I_DOT_2_LSB = PI_TWO_N43
- constexpr double OMEGA_DOT_3_LSB = PI_TWO_N43

- constexpr double DELTA_N_3_LSB = PI_TWO_N43
- constexpr double C_UC_3_LSB = TWO_N29
- constexpr double C_US_3_LSB = TWO_N29
- constexpr double C_RC_3_LSB = TWO_N5
- constexpr double C_RS_3_LSB = TWO_N5
- constexpr double C_IC_4_LSB = TWO_N29
- constexpr double C_IS_4_LSB = TWO_N29
- constexpr int32_t T0C_4_LSB = 60
- constexpr double AF0_4_LSB = TWO_N34
- constexpr double AF1_4_LSB = TWO_N46
- constexpr double AF2_4_LSB = TWO_N59
- constexpr double AI0_5_LSB = TWO_N2
- constexpr double AI1_5_LSB = TWO_N8
- constexpr double AI2_5_LSB = TWO_N15
- constexpr double BGD_E1_E5A_5_LSB = TWO_N32
- constexpr double BGD_E1_E5B_5_LSB = TWO_N32
- constexpr double A0_6_LSB = TWO_N30
- constexpr double A1_6_LSB = TWO_N50
- constexpr int32_t T0T_6_LSB = 3600
- constexpr int32_t T0A_7_LSB = 600
- constexpr double DELTA_A_7_LSB = TWO_N9
- constexpr double E_7_LSB = TWO_N16
- constexpr double OMEGA_7_LSB = TWO_N15
- constexpr double DELTA_I_7_LSB = TWO_N14
- constexpr double OMEGA0_7_LSB = TWO_N15
- constexpr double OMEGA_DOT_7_LSB = TWO_N33
- constexpr double M0_7_LSB = TWO_N15
- constexpr double AF0_8_LSB = TWO_N19
- constexpr double AF1_8_LSB = TWO_N38
- constexpr double DELTA_A_8_LSB = TWO_N9
- constexpr double E_8_LSB = TWO_N16
- constexpr double OMEGA_8_LSB = TWO_N15
- constexpr double DELTA_I_8_LSB = TWO_N14
- constexpr double OMEGA0_8_LSB = TWO_N15
- constexpr double OMEGA_DOT_8_LSB = TWO_N33
- constexpr int32_t T0A_9_LSB = 600
- constexpr double M0_9_LSB = TWO_N15
- constexpr double AF0_9_LSB = TWO_N19
- constexpr double AF1_9_LSB = TWO_N38
- constexpr double DELTA_A_9_LSB = TWO_N9
- constexpr double E_9_LSB = TWO_N16
- constexpr double OMEGA_9_LSB = TWO_N15
- constexpr double DELTA_I_9_LSB = TWO_N14
- constexpr double OMEGA0_10_LSB = TWO_N15
- constexpr double OMEGA_DOT_10_LSB = TWO_N33
- constexpr double M0_10_LSB = TWO_N15
- constexpr double AF0_10_LSB = TWO_N19
- constexpr double AF1_10_LSB = TWO_N38
- constexpr double A_0G_10_LSB = TWO_N35
- constexpr double A_1G_10_LSB = TWO_N51
- constexpr int32_t T_0_G_10_LSB = 3600
- constexpr double CED_DeltaAred_LSB = TWO_P8
- constexpr double CED_exred_LSB = TWO_N22
- constexpr double CED_eyred_LSB = TWO_N22
- constexpr double CED_Deltai0red_LSB = TWO_N22

- constexpr double CED_Omega0red_LSB = TWO_N22
- constexpr double CED_lambda0red_LSB = TWO_N22
- constexpr double CED_af0red_LSB = TWO_N26
- constexpr double CED_af1red_LSB = TWO_N35
- constexpr size_t INAV_RS_SUBVECTOR_LENGTH = 15
- constexpr size_t INAV_RS_PARITY_VECTOR_LENGTH = 60
- constexpr size_t INAV_RS_INFO_VECTOR_LENGTH = 58
- constexpr size_t INAV_RS_BUFFER_LENGTH = 118
- constexpr int32_t BITS_IN_OCTET = 8
- constexpr int32_t FIRST_RS_BIT = 7
- constexpr int32_t FIRST_RS_BIT_AFTER_IODNAV = 17
- constexpr int32_t GALILEO_ISM_CRC_DATA_BITS = 96
- constexpr int32_t GALILEO_ISM_CRC_DATA_BYTES = 12
- constexpr char GALILEO_INAV_PLAIN_SSP1 [9] = "00000100"
- constexpr char GALILEO_INAV_PLAIN_SSP2 [9] = "00101011"
- constexpr char GALILEO_INAV_PLAIN_SSP3 [9] = "00101111"
- constexpr char GALILEO_INAV_ENCODED_SSP1 [17] = "1110100100100101"
- constexpr char GALILEO_INAV_ENCODED_SSP2 [17] = "0110110001001110"
- constexpr char GALILEO_INAV_ENCODED_SSP3 [17] = "1101000000111110"
- constexpr size_t SIZE_DSM_BLOCKS_BYTES = 13
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_1
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_2
- const std::unordered_map< uint8_t, std::pair< uint16_t, uint16_t > > OSNMA_TABLE_3
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_5
- const std::unordered_map< std::string, uint16_t > OSNMA_TABLE_6
- const std::unordered_map< uint8_t, std::pair< uint16_t, uint16_t > > OSNMA_TABLE_7
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_8
- const std::unordered_map< uint8_t, uint16_t > OSNMA_TABLE_10
- const std::unordered_map< uint8_t, uint8_t > OSNMA_TABLE_11
- const std::unordered_map< std::string, uint16_t > OSNMA_TABLE_15
- const std::unordered_map< uint8_t, Mack_lookup > OSNMA_TABLE_16
- constexpr double GLONASS_F_M_A = 0.35e9

  Gravitational constant of atmosphere [m^3/s^2].
- constexpr double GLONASS_SEMI_MAJOR_AXIS = 6378136

  Semi-major axis of Earth [m].
- constexpr double GLONASS_FLATTENING = 1.0 / 29825784.0

  Flattening parameter.
- constexpr double GLONASS_GRAVITY = 97803284.0

  Equatorial acceleration of gravity [mGal].
- constexpr double GLONASS_GRAVITY_CORRECTION = 0.87

  Correction to acceleration of gravity at sea-level due to Atmosphere[uGal].
- constexpr double GLONASS_J2 = 1082625.75e-9

  Second zonal harmonic of the geopotential.
- constexpr double GLONASS_J4 = -2370.89e-9

  Fourth zonal harmonic of the geopotential.
- constexpr double GLONASS_J6 = 6.08e-9

  Sixth zonal harmonic of the geopotential.
- constexpr double GLONASS_J8 = 1.40e-11

  Eighth zonal harmonic of the geopotential.
- constexpr double GLONASS_U0 = 62636861.4

  Normal potential at surface of common terrestrial ellipsoid [m^2/s^2].
- constexpr double GLONASS_C20 = -1082.63e-6

  Second zonal coefficient of spherical harmonic expansion.

- constexpr double GLONASS_EARTH_RADIUS = 6378.136

  Equatorial radius of Earth [km].
- constexpr double GLONASS_EARTH_INCLINATION = 0.000409148809899e3

  Mean inclination of ecliptic to equator (23 deg 26 min 33 sec) [rad].
- constexpr double GLONASS_TAU_0 = -0.005835151531174e3

  (-334 deg 19 min 46.40 sec) [rad];
- constexpr double GLONASS_TAU_1 = 0.071018041257371e3

  (4069 deg 02 min 02.52 sec) [rad];
- constexpr double GLONASS_MOON_Q0 = -0.001115184961435e3

  (-63 deg 53 min 43.41 sec) [rad]
- constexpr double GLONASS_MOON_Q1 = 8.328691103668023e3

  (477198 deg 50 min 56.79 sec) [rad]
- constexpr double GLONASS_MOON_OMEGA_0 = 0.004523601514852e3

  (259 deg 10 min 59.79 sec) [rad]
- constexpr double GLONASS_MOON_OMEGA_1 = -0.033757146246552e3

  (-1934 deg 08 min 31.23 sec) [rad]
- constexpr double GLONASS_MOON_GM = 4902.835

  Lunar gravitational constant [km^3/s^2].
- constexpr double GLONASS_MOON_SEMI_MAJOR_AXIS = 3.84385243e5

  Semi-major axis of lunar orbit [km];.
- constexpr double GLONASS_MOON_ECCENTRICITY = 0.054900489

  Eccentricity of lunar orbit.
- constexpr double GLONASS_MOON_INCLINATION = 0.000089803977407e3

  Inclination of lunar orbit to ecliptic plane (5 deg 08 min 43.4 sec) [rad].
- constexpr double GLONASS_SUN_OMEGA = 0.004908229466869e3

  TODO What is this operation in the seconds with T?(281 deg 13 min 15.0 + 6189.03 x T sec) [rad].
- constexpr double GLONASS_SUN_Q0 = 0.006256583774423e3

  (358 deg 28 min 33.04 sec) [rad]
- constexpr double GLONASS_SUN_Q1 = 0e3

  TODO Why is the value greater than 60?(129596579.10 sec) [rad].
- constexpr double GLONASS_SUN_GM = 0.1325263e12

  Solar gravitational constant [km^3/s^2].
- constexpr double GLONASS_SUN_SEMI_MAJOR_AXIS = 1.49598e8

  Semi-major axis of solar orbit [km];.
- constexpr double GLONASS_SUN_ECCENTRICITY = 0.016719

  Eccentricity of solar orbit.
- constexpr double GLONASS_L2_CA_FREQ_HZ = FREQ2_GLO

  L2 [Hz].
- constexpr double GLONASS_L2_CA_DFREQ_HZ = DFRQ2_GLO

  Freq Bias for GLONASS L1 [Hz].
- constexpr double GLONASS_L2_CA_CODE_RATE_CPS = 0.511e6

  GLONASS L1 C/A code rate [chips/s].
- constexpr double GLONASS_L2_CA_CODE_LENGTH_CHIPS = 511.0

  GLONASS L1 C/A code length [chips].
- constexpr double GLONASS_L2_CA_CODE_PERIOD_S = 0.001

  GLONASS L1 C/A code period [seconds].
- constexpr double GLONASS_L2_CA_SYMBOL_PERIOD_S = 0.01

  GLONASS L1 C/A symbol period [seconds].
- constexpr double GLONASS_L2_CA_CHIP_PERIOD_S = 1.9569e-06

  GLONASS L1 C/A chip period [seconds].
- constexpr double GLONASS_L2_CA_SYMBOL_RATE_BPS = 1000.0

- constexpr double GLONASS_L1_CA_FREQ_HZ = FREQ1_GLO

  L1 [Hz].
- constexpr double GLONASS_L1_CA_DFREQ_HZ = DFRQ1_GLO

  Freq Bias for GLONASS L1 [Hz].
- constexpr double GLONASS_L1_CA_CODE_RATE_CPS = 0.511e6

  GLONASS L1 C/A code rate [chips/s].
- constexpr double GLONASS_L1_CA_CODE_LENGTH_CHIPS = 511.0

  GLONASS L1 C/A code length [chips].
- constexpr double GLONASS_L1_CA_CODE_PERIOD_S = 0.001

  GLONASS L1 C/A code period [seconds].
- constexpr double GLONASS_L1_CA_SYMBOL_PERIOD_S = 0.01

  GLONASS L1 C/A symbol period [seconds].
- constexpr double GLONASS_L1_CA_CHIP_PERIOD_S = 1.9569e-06

  GLONASS L1 C/A chip period [seconds].
- constexpr double GLONASS_L1_CA_SYMBOL_RATE_BPS = 1000.0
- constexpr uint32_t GLONASS_L1_CA_CODE_PERIOD_MS = 1
- constexpr uint32_t GLONASS_L2_CA_CODE_PERIOD_MS = 1
- constexpr int32_t GLONASS_CA_NBR_SATS = 24
- constexpr int32_t GLONASS_L1_CA_HISTORY_DEEP = 100
- constexpr char GLONASS_GNAV_PREAMBLE_STR [301] = "1111111111111111111111111111111111111111111111111111111100
- constexpr double GLONASS_GNAV_PREAMBLE_DURATION_S = 0.300
- constexpr double GLONASS_GNAV_SECONDS_PER_WEEK = 604800.0
- constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_BITS = 30
- constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS = 300
- constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS = 2000
- constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_BITS = 200
- constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND = 50

  NAV message bit rate [bits/s].
- constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT = 10
- constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT = 10
- constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND = GLONASS_GNAV_TELEMETRY_RATE_B
  ∗ GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT

  NAV message bit rate [symbols/s].
- constexpr int32_t GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE = 2000
- constexpr int32_t GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE = 200

  Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
- constexpr int32_t GLONASS_GNAV_STRING_BITS = 85

  Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
- constexpr int32_t GLONASS_GNAV_HAMMING_CODE_BITS = 8

  Number of bits in hamming code sequence of GNAV message.
- constexpr int32_t GLONASS_GNAV_DATA_SYMBOLS = 1700
- constexpr double GLONASS_LEAP_SECONDS [19][7]

  Record of leap seconds definition for GLOT to GPST conversion and vice versa.
- const std::map< uint32_t, int32_t > GLONASS_PRN
- const std::vector< int32_t > GLONASS_GNAV_CRC_I_INDEX {9, 10, 12, 13, 15, 17, 19, 20, 22, 24, 26, 28, 30, 32, 34, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84}
- const std::vector< int32_t > GLONASS_GNAV_CRC_J_INDEX {9, 11, 12, 14, 15, 18, 19, 21, 22, 25, 26, 29, 30, 33, 34, 36, 37, 40, 41, 44, 45, 48, 49, 52, 53, 56, 57, 60, 61, 64, 65, 67, 68, 71, 72, 75, 76, 79, 80, 83, 84}
- const std::vector< int32_t > GLONASS_GNAV_CRC_K_INDEX {10, 11, 12, 16, 17, 18, 19, 23, 24, 25, 26, 31, 32, 33, 34, 38, 39, 40, 41, 46, 47, 48, 49, 54, 55, 56, 57, 62, 63, 64, 65, 69, 70, 71, 72, 77, 78, 79, 80, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_L_INDEX {13, 14, 15, 16, 17, 18, 19, 27, 28, 29, 30, 31, 32, 33, 34, 42, 43, 44, 45, 46, 47, 48, 49, 58, 59, 60, 61, 62, 63, 64, 65, 73, 74, 75, 76, 77, 78, 79, 80}

- const std::vector< int32_t > GLONASS_GNAV_CRC_M_INDEX {20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_N_INDEX {35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65}
- const std::vector< int32_t > GLONASS_GNAV_CRC_P_INDEX {66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_Q_INDEX {9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_ECC_LOCATOR {0, 0, 1, 8, 2, 9, 10, 11, 3, 12, 13, 14, 15, 16, 17, 18, 4, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 5, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 6, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84}
- constexpr double FREQ1 = 1.57542e9

  L1/E1 frequency (Hz).
- constexpr double FREQ2 = 1.22760e9

  L2 frequency (Hz).
- constexpr double FREQ5 = 1.17645e9

  L5/E5a frequency (Hz).
- constexpr double FREQ6 = 1.27875e9

  E6/LEX frequency (Hz).
- constexpr double FREQ7 = 1.20714e9

  E5b frequency (Hz).
- constexpr double FREQ8 = 1.191795e9

  E5a+b frequency (Hz).
- constexpr double FREQ9 = 2.492028e9

  S frequency (Hz).
- constexpr double FREQ1_GLO = 1.60200e9

  GLONASS G1 base frequency (Hz).
- constexpr double DFRQ1_GLO = 0.56250e6

  GLONASS G1 bias frequency (Hz/n).
- constexpr double FREQ2_GLO = 1.24600e9

  GLONASS G2 base frequency (Hz).
- constexpr double DFRQ2_GLO = 0.43750e6

  GLONASS G2 bias frequency (Hz/n).
- constexpr double FREQ3_GLO = 1.202025e9

  GLONASS G3 frequency (Hz).
- constexpr double FREQ1_BDS = 1.561098e9

  BeiDou B1 frequency (Hz).
- constexpr double FREQ2_BDS = 1.20714e9

  BeiDou B2 frequency (Hz).
- constexpr double FREQ3_BDS = 1.26852e9

  BeiDou B3 frequency (Hz).
- const std::unordered_map< std::string, double > SIGNAL_FREQ_MAP
- constexpr uint32_t CODE_NONE = 0

  obs code: none or unknown
- constexpr uint32_t CODE_L1C = 1

  obs code: L1C/A,G1C/A,E1C (GPS,GLO,GAL,QZS,SBS)
- constexpr uint32_t CODE_L1P = 2

  obs code: L1P,G1P (GPS,GLO)
- constexpr uint32_t CODE_L1W = 3

obs code: L1 Z-track (GPS)

- constexpr uint32_t CODE_L1Y = 4

  obs code: L1Y (GPS)
- constexpr uint32_t CODE_L1M = 5

  obs code: L1M (GPS)
- constexpr uint32_t CODE_L1N = 6

  obs code: L1codeless (GPS)
- constexpr uint32_t CODE_L1S = 7

  obs code: L1C(D) (GPS,QZS)
- constexpr uint32_t CODE_L1L = 8

  obs code: L1C(P) (GPS,QZS)
- constexpr uint32_t CODE_L1E = 9

  (not used)
- constexpr uint32_t CODE_L1A = 10

  obs code: E1A (GAL)
- constexpr uint32_t CODE_L1B = 11

  obs code: E1B (GAL)
- constexpr uint32_t CODE_L1X = 12

  obs code: E1B+C,L1C(D+P) (GAL,QZS)
- constexpr uint32_t CODE_L1Z = 13

  obs code: E1A+B+C,L1SAIF (GAL,QZS)
- constexpr uint32_t CODE_L2C = 14

  obs code: L2C/A,G1C/A (GPS,GLO)
- constexpr uint32_t CODE_L2D = 15

  obs code: L2 L1C/A-(P2-P1) (GPS)
- constexpr uint32_t CODE_L2S = 16

  obs code: L2C(M) (GPS,QZS)
- constexpr uint32_t CODE_L2L = 17

  obs code: L2C(L) (GPS,QZS)
- constexpr uint32_t CODE_L2X = 18

  obs code: L2C(M+L),B1I+Q (GPS,QZS,BDS)
- constexpr uint32_t CODE_L2P = 19

  obs code: L2P,G2P (GPS,GLO)
- constexpr uint32_t CODE_L2W = 20

  obs code: L2 Z-track (GPS)
- constexpr uint32_t CODE_L2Y = 21

  obs code: L2Y (GPS)
- constexpr uint32_t CODE_L2M = 22

  obs code: L2M (GPS)
- constexpr uint32_t CODE_L2N = 23

  obs code: L2codeless (GPS)
- constexpr uint32_t CODE_L5I = 24

  obs code: L5/E5aI (GPS,GAL,QZS,SBS)
- constexpr uint32_t CODE_L5Q = 25

  obs code: L5/E5aQ (GPS,GAL,QZS,SBS)
- constexpr uint32_t CODE_L5X = 26

  obs code: L5/E5aI+Q/L5B+C (GPS,GAL,QZS,IRN,SBS)
- constexpr uint32_t CODE_L7I = 27

  obs code: E5bI,B2I (GAL,BDS)
- constexpr uint32_t CODE_L7Q = 28

  obs code: E5bQ,B2Q (GAL,BDS)

- constexpr uint32_t CODE_L7X = 29
  obs code: E5bI+Q,B2I+Q (GAL,BDS)
- constexpr uint32_t CODE_L6A = 30
  obs code: E6A (GAL)
- constexpr uint32_t CODE_L6B = 31
  obs code: E6B (GAL)
- constexpr uint32_t CODE_L6C = 32
  obs code: E6C (GAL)
- constexpr uint32_t CODE_L6X = 33
  obs code: E6B+C,LEXS+L,B3I+Q (GAL,QZS,BDS)
- constexpr uint32_t CODE_L6Z = 34
  obs code: E6A+B+C (GAL)
- constexpr uint32_t CODE_L6S = 35
  obs code: LEXS (QZS)
- constexpr uint32_t CODE_L6L = 36
  obs code: LEXL (QZS)
- constexpr uint32_t CODE_L8I = 37
  obs code: E5(a+b)I (GAL)
- constexpr uint32_t CODE_L8Q = 38
  obs code: E5(a+b)Q (GAL)
- constexpr uint32_t CODE_L8X = 39
  obs code: E5(a+b)I+Q (GAL)
- constexpr uint32_t CODE_L2I = 40
  obs code: B1I (BDS)
- constexpr uint32_t CODE_L2Q = 41
  obs code: B1Q (BDS)
- constexpr uint32_t CODE_L6I = 42
  obs code: B3I (BDS)
- constexpr uint32_t CODE_L6Q = 43
  obs code: B3Q (BDS)
- constexpr uint32_t CODE_L3I = 44
  obs code: G3I (GLO)
- constexpr uint32_t CODE_L3Q = 45
  obs code: G3Q (GLO)
- constexpr uint32_t CODE_L3X = 46
  obs code: G3I+Q (GLO)
- constexpr uint32_t CODE_L1I = 47
  obs code: B1I (BDS)
- constexpr uint32_t CODE_L1Q = 48
  obs code: B1Q (BDS)
- constexpr uint32_t CODE_L5A = 49
  obs code: L5A SPS (IRN)
- constexpr uint32_t CODE_L5B = 50
  obs code: L5B RS(D) (IRN)
- constexpr uint32_t CODE_L5C = 51
  obs code: L5C RS(P) (IRN)
- constexpr uint32_t CODE_L9A = 52
  obs code: SA SPS (IRN)
- constexpr uint32_t CODE_L9B = 53
  obs code: SB RS(D) (IRN)
- constexpr uint32_t CODE_L9C = 54

obs code: SC RS(P) (IRN)

- constexpr uint32_t CODE_L9X = 55

obs code: SB+C (IRN)

- constexpr uint32_t CODE_L1D = 56

obs code: B1C data (BDS)

- constexpr uint32_t CODE_L5D = 57

obs code: B2a data (BDS)

- constexpr int32_t MAXCODE = 57

max number of obs code

- constexpr int32_t GPS_CNAV_DATA_PAGE_BITS = 300
- constexpr int32_t CNAV_TOW_LSB = 6
- constexpr int32_t CNAV_TOP1_LSB = 300
- constexpr int32_t CNAV_TOE1_LSB = 300
- constexpr double CNAV_DELTA_A_LSB = TWO_N9
- constexpr double CNAV_A_DOT_LSB = TWO_N21
- constexpr double CNAV_DELTA_N0_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI
- constexpr double CNAV_M0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_E_ECCENTRICITY_LSB = TWO_N34
- constexpr double CNAV_OMEGA_LSB = TWO_N32 * GNSS_PI
- constexpr int32_t CNAV_TOE2_LSB = 300
- constexpr double CNAV_OMEGA0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_I0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_DELTA_OMEGA_DOT_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_I0_DOT_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_CIS_LSB = TWO_N30
- constexpr double CNAV_CIC_LSB = TWO_N30
- constexpr double CNAV_CRS_LSB = TWO_N8
- constexpr double CNAV_CRC_LSB = TWO_N8
- constexpr double CNAV_CUS_LSB = TWO_N30
- constexpr double CNAV_CUC_LSB = TWO_N30
- constexpr int32_t CNAV_TOP2_LSB = 300
- constexpr int32_t CNAV_TOC_LSB = 300
- constexpr double CNAV_AF0_LSB = TWO_N35
- constexpr double CNAV_AF1_LSB = TWO_N48
- constexpr double CNAV_AF2_LSB = TWO_N60
- constexpr double CNAV_TGD_LSB = TWO_N35
- constexpr double CNAV_ISCL1_LSB = TWO_N35
- constexpr double CNAV_ISCL2_LSB = TWO_N35
- constexpr double CNAV_ISCL5I_LSB = TWO_N35
- constexpr double CNAV_ISCL5Q_LSB = TWO_N35
- constexpr double CNAV_ALPHA0_LSB = TWO_N30
- constexpr double CNAV_ALPHA1_LSB = TWO_N27
- constexpr double CNAV_ALPHA2_LSB = TWO_N24
- constexpr double CNAV_ALPHA3_LSB = TWO_N24
- constexpr double CNAV_BETA0_LSB = TWO_P11
- constexpr double CNAV_BETA1_LSB = TWO_P14
- constexpr double CNAV_BETA2_LSB = TWO_P16
- constexpr double CNAV_BETA3_LSB = TWO_P16
- constexpr int32_t CNAV_T_EOP_LSB = TWO_P4
- constexpr double CNAV_PM_X_LSB = TWO_N20
- constexpr double CNAV_PM_X_DOT_LSB = TWO_N21
- constexpr double CNAV_PM_Y_LSB = TWO_N20
- constexpr double CNAV_PM_Y_DOT_LSB = TWO_N21

- constexpr double CNAV_DELTA_UT1_GPS_LSB = TWO_N23
- constexpr double CNAV_DELTA_UT1_GPS_DOT_LSB = TWO_N25
- constexpr double CNAV_A0_LSB = TWO_N35
- constexpr double CNAV_A1_LSB = TWO_N51
- constexpr double CNAV_A2_LSB = TWO_N68
- constexpr int32_t CNAV_DELTA_TLS_LSB = 1
- constexpr int32_t CNAV_TOT_LSB = TWO_P4
- constexpr int32_t CNAV_WN_OT_LSB = 1
- constexpr int32_t CNAV_WN_LSF_LSB = 1
- constexpr int32_t CNAV_DN_LSB = 1
- constexpr int32_t CNAV_DELTA_TLSF_LSB = 1
- constexpr double CNAV_A_REF = 26559710.0
- constexpr double CNAV_OMEGA_DOT_REF = -2.6e-9
- constexpr double CNAV_QZSS_A_REF = 42164200.0
- constexpr double GPS_L1_FREQ_HZ = FREQ1

  L1 [Hz].
- constexpr double GPS_L1_CA_CODE_RATE_CPS = 1.023e6

  GPS L1 C/A code rate [chips/s].
- constexpr double GPS_L1_CA_CODE_LENGTH_CHIPS = 1023.0

  GPS L1 C/A code length [chips].
- constexpr double GPS_L1_CA_CODE_PERIOD_S = 0.001

  GPS L1 C/A code period [seconds].
- constexpr double GPS_L1_CA_CHIP_PERIOD_S = 9.7752e-07

  GPS L1 C/A chip period [seconds].
- constexpr uint32_t GPS_L1_CA_CODE_PERIOD_MS = 1U

  GPS L1 C/A code period [ms].
- constexpr uint32_t GPS_L1_CA_BIT_PERIOD_MS = 20U

  GPS L1 C/A bit period [ms].
- constexpr double MAX_TOA_DELAY_MS = 20.0

  Maximum Time-Of-Arrival (TOA) difference between satellites for a receiver operated on Earth surface is 20 ms.
- constexpr uint32_t GPS_L1_CA_OPT_ACQ_FS_SPS = 2000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L1_CA_HISTORY_DEEP = 100
- constexpr double GPS_CA_PREAMBLE_DURATION_S = 0.160
- constexpr int32_t GPS_CA_PREAMBLE_LENGTH_BITS = 8
- constexpr int32_t GPS_CA_PREAMBLE_LENGTH_SYMBOLS = 160
- constexpr int32_t GPS_CA_PREAMBLE_DURATION_MS = 160
- constexpr int32_t GPS_CA_TELEMETRY_RATE_BITS_SECOND = 50

  NAV message bit rate [bits/s].
- constexpr int32_t GPS_CA_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND = GPS_CA_TELEMETRY_RATE_BITS_SECOND ∗ GPS_CA_TELEMETRY_SYMBOLS_PER_BIT

  NAV message bit rate [symbols/s].
- constexpr int32_t GPS_WORD_LENGTH = 4

  CRC + GPS WORD (-2 -1 0 ... 29) Bits = 4 bytes.
- constexpr int32_t GPS_SUBFRAME_LENGTH = 40

  GPS_WORD_LENGTH x 10 = 40 bytes.
- constexpr int32_t GPS_SUBFRAME_BITS = 300

  Number of bits per subframe in the NAV message [bits].
- constexpr int32_t GPS_SUBFRAME_SECONDS = 6

  Subframe duration [seconds].
- constexpr int32_t GPS_SUBFRAME_MS = 6000

Subframe duration [seconds].

- constexpr int32_t GPS_WORD_BITS = 30

  Number of bits per word in the NAV message [bits].
- constexpr int32_t GPS_LNAV_DATA_ID = 1

  IS-GPS-200N LNAV Data ID number 2, encoded as binary 01.
- constexpr char GPS_CA_PREAMBLE [9] = "10001011"
- constexpr char GPS_CA_PREAMBLE_SYMBOLS_STR [161] = "11111111111111111111000000000000000000000000000000
- constexpr double T_GD_LSB = TWO_N31
- constexpr int32_t T_OC_LSB = static_cast<int32_t>(TWO_P4)
- constexpr double A_F2_LSB = TWO_N55
- constexpr double A_F1_LSB = TWO_N43
- constexpr double A_F0_LSB = TWO_N31
- constexpr double C_RS_LSB = TWO_N5
- constexpr double DELTA_N_LSB = PI_TWO_N43
- constexpr double M_0_LSB = PI_TWO_N31
- constexpr double C_UC_LSB = TWO_N29
- constexpr double ECCENTRICITY_LSB = TWO_N33
- constexpr double C_US_LSB = TWO_N29
- constexpr double SQRT_A_LSB = TWO_N19
- constexpr int32_t T_OE_LSB = static_cast<int32_t>(TWO_P4)
- constexpr int32_t AODO_LSB = 900
- constexpr double C_IC_LSB = TWO_N29
- constexpr double OMEGA_0_LSB = PI_TWO_N31
- constexpr double C_IS_LSB = TWO_N29
- constexpr double I_0_LSB = PI_TWO_N31
- constexpr double C_RC_LSB = TWO_N5
- constexpr double OMEGA_LSB = PI_TWO_N31
- constexpr double OMEGA_DOT_LSB = PI_TWO_N43
- constexpr double I_DOT_LSB = PI_TWO_N43
- constexpr double ALPHA_0_LSB = TWO_N30
- constexpr double ALPHA_1_LSB = TWO_N27
- constexpr double ALPHA_2_LSB = TWO_N24
- constexpr double ALPHA_3_LSB = TWO_N24
- constexpr double BETA_0_LSB = TWO_P11
- constexpr double BETA_1_LSB = TWO_P14
- constexpr double BETA_2_LSB = TWO_P16
- constexpr double BETA_3_LSB = TWO_P16
- constexpr double A_1_LSB = TWO_N50
- constexpr double A_0_LSB = TWO_N30
- constexpr double T_OT_LSB = TWO_P12
- constexpr double WN_T_LSB = 1
- constexpr double DELTAT_LS_LSB = 1
- constexpr double WN_LSF_LSB = 1
- constexpr double DN_LSB = 1
- constexpr double DELTAT_LSF_LSB = 1
- constexpr int32_t GPS_SV_CONFIG_AS_FLAG_MASK = 0x8

  A-S flag (1 = ON).
- constexpr int32_t GPS_SV_CONFIG_CODE_MASK = 0x7

  SV configuration.
- constexpr int32_t GPS_SV_CONFIG_UNKNOWN = 0

  000: no information
- constexpr int32_t GPS_SV_CONFIG_BLOCK_IIR_M = 2

  010: first with L2C
- constexpr int32_t GPS_SV_CONFIG_BLOCK_IIF = 3

011: first with L5

- constexpr int32_t T_OA_LSB = TWO_P12
- constexpr double ALM_ECC_LSB = TWO_N21
- constexpr int32_t ALM_TOA_LSB = static_cast<int32_t>(TWO_P12)
- constexpr double ALM_DELTAI_LSB = TWO_N19
- constexpr double ALM_OMEGADOT_LSB = TWO_N38
- constexpr double ALM_SQUAREA_LSB = TWO_N11
- constexpr double ALM_OMEGAZERO_LSB = TWO_N23
- constexpr double ALM_OMEGA_LSB = TWO_N23
- constexpr double ALM_MZERO_LSB = TWO_N23
- constexpr double ALM_AF0_LSB = TWO_N20
- constexpr double ALM_AF1_LSB = TWO_N38
- constexpr double GPS_L2_FREQ_HZ = FREQ2

  L2 [Hz].
- constexpr double GPS_L2_L_PERIOD_S = 1.5

  GPS L2 L code period [seconds].
- constexpr double GPS_L2_M_CODE_RATE_CPS = 0.5115e6

  GPS L2 M code rate [chips/s].
- constexpr double GPS_L2_M_CODE_RATE_L_ZEROED_CPS = 1.023e6

  GPS L2 M code rate with L2 M Zeroed [chips/s].
- constexpr double GPS_L2_M_PERIOD_S = 0.02

  GPS L2 M code period [seconds].
- constexpr double GPS_L2_L_CODE_RATE_CPS = 0.5115e6

  GPS L2 L code rate [chips/s].
- constexpr uint32_t GPS_L2_M_CODE_PERIOD_MS = 20

  GPS L2 L code period [ms].
- constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS = 10230

  GPS L2 M code length [chips].
- constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED = 2 ∗ GPS_L2_M_CODE_LENGTH_CHIPS

  GPS L2 M code length with L2 CL zeroed [chips].
- constexpr int32_t GPS_L2_L_CODE_LENGTH_CHIPS = 767250

  GPS L2 L code length [chips].
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_BITS = 300

  GPS L2 CNAV page length, including preamble and CRC [bits].
- constexpr int32_t GPS_L2_SYMBOLS_PER_BIT = 2
- constexpr int32_t GPS_L2_SAMPLES_PER_SYMBOL = 1
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_SYMBOLS = 600
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_DURATION_S = 12
- constexpr int32_t GPS_L2C_HISTORY_DEEP = 5
- constexpr uint32_t GPS_L2C_OPT_ACQ_FS_SPS = 2000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L2C_M_INIT_REG [115]
- constexpr double GPS_L5_FREQ_HZ = FREQ5

  L5 [Hz].
- constexpr double GPS_L5I_CODE_RATE_CPS = 10.23e6

  GPS L5I code rate [chips/s].
- constexpr double GPS_L5I_PERIOD_S = 0.001

  GPS L5I code period [seconds].
- constexpr double GPS_L5I_SYMBOL_PERIOD_S = 0.01

  GPS L5I symbol period [seconds].
- constexpr double GPS_L5Q_CODE_RATE_CPS = 10.23e6

  GPS L5Q code rate [chips/s].

- constexpr double GPS_L5Q_PERIOD_S = 0.001

  GPS L5Q code period [seconds].
- constexpr int32_t GPS_L5Q_CODE_LENGTH_CHIPS = 10230

  GPS L5Q code length [chips].
- constexpr int32_t GPS_L5I_CODE_LENGTH_CHIPS = 10230

  GPS L5I code length [chips].
- constexpr int32_t GPS_L5I_PERIOD_MS = 1

  GPS L5I code period [ms].
- constexpr int32_t GPS_L5I_SYMBOL_PERIOD_MS = 10

  GPS L5I symbol period [ms].
- constexpr int32_t GPS_L5_HISTORY_DEEP = 5
- constexpr uint32_t GPS_L5_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L5I_INIT_REG [210]
- constexpr int32_t GPS_L5Q_INIT_REG [210]
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_BITS = 300

  GPS L5 CNAV page length, including preamble and CRC [bits].
- constexpr int32_t GPS_L5_SYMBOLS_PER_BIT = 2
- constexpr int32_t GPS_L5_SAMPLES_PER_SYMBOL = 10
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_SYMBOLS = 600
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_DURATION_S = 6
- constexpr int32_t GPS_L5I_NH_CODE_LENGTH = 10
- constexpr int32_t GPS_L5I_NH_CODE [10] = {0, 0, 0, 0, 1, 1, 0, 1, 0, 1}
- constexpr int32_t GPS_L5Q_NH_CODE_LENGTH = 20
- constexpr int32_t GPS_L5Q_NH_CODE [20] = {0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0}
- constexpr char GPS_L5I_NH_CODE_STR [11] = "0000110101"
- constexpr char GPS_L5Q_NH_CODE_STR [21] = "00000100110101001110"
- constexpr double GNSS_OMEGA_EARTH_DOT = 7.2921151467e-5

  Default Earth rotation rate, [rad/s].
- constexpr double SPEED_OF_LIGHT_M_S = 299792458.0

  Speed of light in vacuum [m/s].
- constexpr double SPEED_OF_LIGHT_M_MS = 299792.4580

  Speed of light in vacuum [m/ms].
- constexpr double GPS_GM = 3.986005e14

  Universal gravitational constant times the mass of the Earth, [m^3/s^2] IS-GPS-200M, 20.3.3.3.3.1.
- constexpr double GPS_F = -4.442807633e-10

  Constant, [s/(m)^(1/2)], IS-GPS-200M, 20.3.3.3.3.1.
- constexpr double GALILEO_GM = 3.986004418e14

  Geocentric gravitational constant[m^3/s^2], OS SIS ICD v2.0, pag. 44.
- constexpr double GALILEO_F = -4.442807309e-10

  Constant, [s/(m)^(1/2)]. OS SIS ICD v2.0, pag. 47.
- constexpr double GLONASS_OMEGA_EARTH_DOT = 7.292115e-5

  Earth rotation rate, [rad/s] ICD L1, L2 GLONASS Edition 5.1 2008 pag. 55.
- constexpr double GLONASS_GM = 398600.44e9

  Universal gravitational constant times the mass of the Earth, [m^3/s^2].
- constexpr double BEIDOU_OMEGA_EARTH_DOT = 7.2921150e-5

  Earth rotation rate, [rad/s] as defined in BDS-SIS-ICD-B1I-3.0 2019-02, pag. 3.
- constexpr double BEIDOU_GM = 3.986004418e14

  Universal gravitational constant times the mass of the Earth, [m^3/s^2] as defined in CGCS2000.
- constexpr double BEIDOU_F = -4.442807309e-10

  Constant, [s/(m)^(1/2)] F=-2(GM)^.5/C^2.
- constexpr double GNSS_PI = 3.1415926535898

pi constant as defined for GNSS

- constexpr double HALF_PI = GNSS_PI / 2.0

  pi/2
- constexpr double TWO_PI = 2.0 ∗ GNSS_PI

  2 ∗ pi
- constexpr double TWO_P3 = 8.0

  2^3
- constexpr double TWO_P4 = 16.0

  2^4
- constexpr double TWO_P8 = 256.0

  2^8
- constexpr double TWO_P11 = 2048.0

  2^11
- constexpr double TWO_P12 = 4096.0

  2^12
- constexpr double TWO_P14 = 16384.0

  2^14
- constexpr double TWO_P16 = 65536.0

  2^16
- constexpr double TWO_P19 = 524288.0

  2^19
- constexpr double TWO_P31 = 2147483648.0

  2^31
- constexpr double TWO_P32 = 4294967296.0

  2^32
- constexpr double TWO_P56 = 7.205759403792794e+016

  2^56
- constexpr double TWO_P57 = 1.441151880758559e+017

  2^57
- constexpr double TWO_N2 = 0.25

  2^-2
- constexpr double TWO_N5 = 0.03125

  2^-5
- constexpr double TWO_N6 = 0.015625

  2^-6
- constexpr double TWO_N8 = 0.00390625

  2^-8
- constexpr double TWO_N9 = 0.001953125

  2^-9
- constexpr double TWO_N10 = 0.0009765625

  2^-10
- constexpr double TWO_N11 = 4.882812500000000e-004

  2^-11
- constexpr double TWO_N14 = 0.00006103515625

  2^-14
- constexpr double TWO_N15 = 3.051757812500000e-005

  2^-15
- constexpr double TWO_N16 = 1.525878906250000e-005

  2^-16
- constexpr double TWO_N17 = 7.629394531250000e-006

  2^-17

- constexpr double TWO_N18 = 3.814697265625000e-006
  2^-18
- constexpr double TWO_N19 = 1.907348632812500e-006
  2^-19
- constexpr double TWO_N20 = 9.536743164062500e-007
  2^-20
- constexpr double TWO_N21 = 4.768371582031250e-007
  2^-21
- constexpr double TWO_N22 = 2.384185791015625e-007
  2^-22
- constexpr double TWO_N23 = 1.192092895507810e-007
  2^-23
- constexpr double TWO_N24 = 5.960464477539063e-008
  2^-24
- constexpr double TWO_N25 = 2.980232238769531e-008
  2^-25
- constexpr double TWO_N26 = 1.490116119384765e-009
  2^-26
- constexpr double TWO_N27 = 7.450580596923828e-009
  2^-27
- constexpr double TWO_N29 = 1.862645149230957e-009
  2^-29
- constexpr double TWO_N30 = 9.313225746154785e-010
  2^-30
- constexpr double TWO_N31 = 4.656612873077393e-010
  2^-31
- constexpr double TWO_N32 = 2.328306436538696e-010
  2^-32
- constexpr double TWO_N33 = 1.164153218269348e-010
  2^-33
- constexpr double TWO_N34 = 5.82076609134674e-011
  2^-34
- constexpr double TWO_N35 = 2.91038304567337e-011
  2^-35
- constexpr double TWO_N38 = 3.637978807091713e-012
  2^-38
- constexpr double TWO_N39 = 1.818989403545856e-012
  2^-39
- constexpr double TWO_N40 = 9.094947017729280e-013
  2^-40
- constexpr double TWO_N43 = 1.136868377216160e-013
  2^-43
- constexpr double TWO_N44 = 5.684341886080802e-14
  2^-44
- constexpr double TWO_N46 = 1.4210854715202e-014
  2^-46
- constexpr double TWO_N48 = 3.552713678800501e-15
  2^-46
- constexpr double TWO_N50 = 8.881784197001252e-016
  2^-50
- constexpr double TWO_N51 = 4.44089209850063e-016

2^-51

- constexpr double TWO_N55 = 2.775557561562891e-017

  2^-55
- constexpr double TWO_N57 = 6.938893903907228e-18

  2^-57
- constexpr double TWO_N59 = 1.73472347597681e-018

  2^-59
- constexpr double TWO_N60 = 8.673617379884036e-19

  2^-60
- constexpr double TWO_N66 = 1.3552527156068805425093160010874271392822265625e-20

  2^-66
- constexpr double TWO_N68 = 3.388131789017201e-21

  2^-68
- constexpr double PI_TWO_N19 = 5.992112452678286e-006

  *Pi*2^-19.*
- constexpr double PI_TWO_N43 = 3.571577341960839e-013

  *Pi*2^-43.*
- constexpr double PI_TWO_N31 = 1.462918079267160e-009

  *Pi*2^-31.*
- constexpr double PI_TWO_N38 = 1.142904749427469e-011

  *Pi*2^-38.*
- constexpr double PI_TWO_N23 = 3.745070282923929e-007

  *Pi*2^-23.*
- constexpr double D2R = GNSS_PI / 180.0

  deg to rad
- constexpr double R2D = 180.0 / GNSS_PI

  rad to deg
- constexpr double SC2RAD = GNSS_PI

  semi-circle to radian (IS-GPS)
- constexpr double AS2R = D2R / 3600.0

  arc sec to radian
- constexpr double AU = 149597870691.0

  1 Astronomical Unit AU (m) distance from Earth to the Sun.
- constexpr const char QZSS_CA_PREAMBLE_SYMBOLS_STR [161] = "11111111111111111110000000000000000000000000
- constexpr const char QZSS_L5Q_NH_CODE_STR [21] = "00000100110101001110"
- constexpr const char QZSS_L5I_NH_CODE_STR [11] = "0000110101"
- constexpr double SBAS_L1_FREQ_HZ = FREQ1

  SBAS L1 carrier frequency [Hz] (same as GPS L1).
- constexpr double SBAS_L1_CODE_RATE_CPS = 1.023e6

  SBAS L1 C/A code chip rate [chips/s].
- constexpr double SBAS_L1_CODE_LENGTH_CHIPS = 1023.0

  SBAS L1 C/A code length [chips].
- constexpr double SBAS_L1_CODE_PERIOD_S = 0.001

  SBAS L1 C/A code period [s].
- constexpr uint32_t SBAS_L1_CODE_PERIOD_MS = 1U

  SBAS L1 C/A code period [ms].
- constexpr uint32_t SBAS_L1_OPT_ACQ_FS_SPS = 2000000U

  Sampling frequency that maximises acquisition SNR [samples/s].
- constexpr uint32_t SBAS_L1_PRN_MIN = 120U

  Minimum SBAS PRN (inclusive).
- constexpr uint32_t SBAS_L1_PRN_MAX = 138U

Maximum SBAS PRN (inclusive).

- constexpr int32_t SBAS_L1_DATA_RATE_BPS = 250

  Raw data bit rate [bits/s].
- constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_RATE_SPS = 500

  Encoded channel symbol rate (rate-1/2 FEC) [symbols/s].
- constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS = 2

  Encoded channel symbol period [ms] (1/500 sps).
- constexpr int32_t SBAS_L1_MSG_LENGTH_BITS = 250

  Information bits per 1-second message frame [bits].
- constexpr int32_t SBAS_L1_MSG_LENGTH_BYTES = 32

  Bytes holding a 250-bit frame with six trailing zero bits.
- constexpr int32_t SBAS_L1_DATA_FIELD_BYTES = 29

  Bytes holding preamble, type, and 212-bit data field.
- constexpr int32_t SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS = 1
- constexpr uint8_t SBAS_L1_PREAMBLE_1 = 0x53U

  First preamble byte (binary: 0101 0011).
- constexpr uint8_t SBAS_L1_PREAMBLE_2 = 0x9AU

  Second preamble byte (binary: 1001 1010).
- constexpr uint8_t SBAS_L1_PREAMBLE_3 = 0xC6U

  Third preamble byte (binary: 1100 0110).

10.2.6.1 Detailed Description

Classes containing info about system parameters for the different GNSS.
GNSS parameters

10.2.6.2 Macro Definition Documentation

10.2.6.2.1 DISPLAY_COLORS

```
#define DISPLAY_COLORS 1
```

Definition at line 29 of file display.h.

10.2.6.2.2 GLONASS_GNAV_PREAMBLE

```
#define GLONASS_GNAV_PREAMBLE
```

Value:

```
{                          \
    1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0}
```

Definition at line 94 of file GLONASS_L1_L2_CA.h.

10.2.6.2.3 GLONASS_GNAV_PREAMBLE_SAMPLES

```
#define GLONASS_GNAV_PREAMBLE_SAMPLES
```

Value:

```
{                                    \
    1, 1, 1, 1, 1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, 1, -1, -1, 1, -1, 1, 1, -1}
```

Definition at line 98 of file GLONASS_L1_L2_CA.h.

10.2.6.3 Typedef Documentation

10.2.6.3.1 BeidouCnav1LdpcGraph

```
using BeidouCnav1LdpcGraph = BeidouLdpcGraph
```

Definition at line 120 of file beidou_cnav1_ldpc.h.

10.2.6.4 Enumeration Type Documentation

10.2.6.4.1 CnavSystem

```
enum class CnavSystem  [strong]
```
Definition at line 40 of file gps_cnav_navigation_message.h.

10.2.6.4.2 Galileo_Nav_Message_Source

```
enum class Galileo_Nav_Message_Source :  uint8_t  [strong]
```
Definition at line 48 of file galileo_ephemeris.h.

10.2.6.4.3 Galileo_Nav_Message_Type

```
enum class Galileo_Nav_Message_Type :  uint8_t  [strong]
```
This class is a storage and orbital model functions for the Galileo SV ephemeris data as described in Galileo ICD paragraph 5.1.1.
(See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_↩OS_SIS_ICD_v2.0.pdf)
Definition at line 40 of file galileo_ephemeris.h.

10.2.6.4.4 LnavSystem

```
enum class LnavSystem  [strong]
```
Definition at line 41 of file gps_navigation_message.h.

10.2.6.5 Function Documentation

10.2.6.5.1 ALPHA_0()

```
const std::vector< std::pair< int32_t, int32_t > > ALPHA_0 (
            {{69, 8}} )
```

Todo read all pages of subframe 4

References ALPHA_0().
Referenced by ALPHA_0().

10.2.6.5.2 beidou_bdgim_delay_m()

```
double beidou_bdgim_delay_m (
            double time_mjd,
            double lat_rad,
            double lon_rad,
            double az_rad,
            double el_rad,
            const double alpha[9],
            double freq_hz)
```
Compute BDGIM slant ionospheric delay [m] at carrier frequency freq_hz.

Parameters

time_mjd	Modified Julian Date (days) of the computation epoch
lat_rad	User geodetic latitude [rad]
lon_rad	User geodetic longitude [rad]
az_rad	Satellite azimuth [rad]
el_rad	Satellite elevation [rad]
alpha	Broadcast BDGIM parameters alpha1..alpha9 [TECu]

freq_hz	Carrier frequency [Hz]

Returns

Slant ionospheric delay [m], or 0 if inputs are invalid

References beidou_bdgim_delay_m().
Referenced by beidou_bdgim_delay_m().

10.2.6.5.3 beidou_bdgim_nearest_odd_hour_mjd()

```
double beidou_bdgim_nearest_odd_hour_mjd (
            double time_mjd)
```

ICD §7.8.2: odd hour of day (01:00 . . . 23:00) nearest to time_mjd [MJD days].
References beidou_bdgim_nearest_odd_hour_mjd().
Referenced by beidou_bdgim_nearest_odd_hour_mjd().

10.2.6.5.4 gps_ref_week_from_config()

```
int32_t gps_ref_week_from_config (
            const std::string & observation_date,
            bool pre_2009_file)
```

Derives the reference GPS week number used to resolve the GPS mod-1024 week-number rollover when post-processing recorded signal files.

Parameters

observation_date	Value of the GNSS-SDR.observation_date configuration option: approximate date of the signal capture, in "YYYY-MM-DD" or "YYYY" format. An empty string means "not set".
pre_2009_file	Value of the deprecated GNSS-SDR.pre_2009_file flag.

Returns

The full GPS week number of the observation date. Each mod-1024 broadcast week number is then resolved to the 1024-week era that places it closest to this reference week (see adjgpsweek()). Returns 0 if neither option is set, which keeps the default behavior of deriving the era from the system clock. If only the deprecated flag is set, it returns week 1535 (mid-point of the 1999-2019 era), which reproduces the legacy fixed +1024 week adjustment, and a one-time deprecation notice is printed.

References gps_ref_week_from_config().
Referenced by gps_ref_week_from_config().

10.2.6.5.5 parse_agnss_ref_location()

```
Agnss_Ref_Location parse_agnss_ref_location (
            const std::string & ref_location_str)  [inline]
```

Parses GNSS-SDR.AGNSS_ref_location ("lat,lon" or "lat lon", degrees). valid is false if the string is empty or not a plausible WGS84 position. Single parsing point for ControlThread and SatelliteVisibility.
Definition at line 74 of file agnss_ref_location.h.
References parse_agnss_ref_location().
Referenced by parse_agnss_ref_location().

10.2.6.5.6 parse_agnss_ref_utc_time()

```
Agnss_Ref_Time parse_agnss_ref_utc_time (
            const std::string & ref_time_str,
            bool * malformed_year = nullptr,
            bool * malformed_format = nullptr)  [inline]
```

Parses GNSS-SDR.AGNSS_ref_utc_time ("DD/MM/YYYY HH:MM:SS", UTC). An empty string is not an error: the host's current time is returned, marked valid. malformed_year and malformed_format (both optional) tell why a non-empty string was rejected. Single parsing point for ControlThread and SatelliteVisibility.
Definition at line 76 of file agnss_ref_time.h.
References parse_agnss_ref_utc_time().
Referenced by parse_agnss_ref_utc_time().

10.2.6.5.7 qzss_l1cb_prn_to_nominal_prn()

```
uint32_t qzss_l1cb_prn_to_nominal_prn (
            uint32_t prn)  [constexpr]
```

Maps QZSS L1 C/B PRNs to the PRN carrying the satellite's nominal PNT signals. L1 C/B is broadcast with a dedicated PRN (203-206), but the transmitting satellite is identified by the PRN of its nominal signals (RINEX 4.00, Table 6: QZSS PRN to RINEX Satellite Identifier): 203->196 (J04), 204->197 (J05), 205->200 (J08), 206->201 (J09). Returns the input PRN unchanged if it is not an L1 C/B alias; in particular, PRN 202 (J10) broadcasts L1 C/B under its own PRN.
Definition at line 75 of file qzss.h.
References qzss_l1cb_prn_to_nominal_prn().
Referenced by qzss_l1cb_prn_to_nominal_prn().

10.2.6.5.8 T_OA()

```
const std::vector< std::pair< int32_t, int32_t > > T_OA (
            {{69, 8}} )
```

Todo read all pages of subframe 5

References T_OA().
Referenced by T_OA().

10.2.6.6 Variable Documentation

10.2.6.6.1 A0_6_LSB

```
double A0_6_LSB = TWO_N30  [constexpr]
```

Definition at line 148 of file Galileo_INAV.h.

10.2.6.6.2 A1_6_LSB

```
double A1_6_LSB = TWO_N50  [constexpr]
```

Definition at line 150 of file Galileo_INAV.h.

10.2.6.6.3 A_0_LSB

```
double A_0_LSB = TWO_N30  [constexpr]
```

Definition at line 172 of file GPS_L1_CA.h.

10.2.6.6.4 A_0G_10_LSB

```
double A_0G_10_LSB = TWO_N35  [constexpr]
```

Definition at line 245 of file Galileo_INAV.h.

10.2.6.6.5 A_1_LSB

```
double A_1_LSB = TWO_N50  [constexpr]
```

Definition at line 170 of file GPS_L1_CA.h.

10.2.6.6.6 A_1_LSB_GAL

```
double A_1_LSB_GAL = TWO_N19  [constexpr]
```

Definition at line 66 of file Galileo_INAV.h.

10.2.6.6.7 A_1G_10_LSB

```
double A_1G_10_LSB = TWO_N51  [constexpr]
```
Definition at line 247 of file Galileo_INAV.h.

10.2.6.6.8 A_F0_LSB

```
double A_F0_LSB = TWO_N31  [constexpr]
```
Definition at line 103 of file GPS_L1_CA.h.

10.2.6.6.9 A_F1_LSB

```
double A_F1_LSB = TWO_N43  [constexpr]
```
Definition at line 101 of file GPS_L1_CA.h.

10.2.6.6.10 A_F2_LSB

```
double A_F2_LSB = TWO_N55  [constexpr]
```
Definition at line 99 of file GPS_L1_CA.h.

10.2.6.6.11 AF0_10_LSB

```
double AF0_10_LSB = TWO_N19  [constexpr]
```
Definition at line 239 of file Galileo_INAV.h.

10.2.6.6.12 AF0_4_LSB

```
double AF0_4_LSB = TWO_N34  [constexpr]
```
Definition at line 109 of file Galileo_INAV.h.

10.2.6.6.13 AF0_8_LSB

```
double AF0_8_LSB = TWO_N19  [constexpr]
```
Definition at line 186 of file Galileo_INAV.h.

10.2.6.6.14 AF0_9_LSB

```
double AF0_9_LSB = TWO_N19  [constexpr]
```
Definition at line 214 of file Galileo_INAV.h.

10.2.6.6.15 AF1_10_LSB

```
double AF1_10_LSB = TWO_N38  [constexpr]
```
Definition at line 241 of file Galileo_INAV.h.

10.2.6.6.16 AF1_4_LSB

```
double AF1_4_LSB = TWO_N46  [constexpr]
```
Definition at line 111 of file Galileo_INAV.h.

10.2.6.6.17 AF1_8_LSB

```
double AF1_8_LSB = TWO_N38  [constexpr]
```
Definition at line 188 of file Galileo_INAV.h.

10.2.6.6.18 AF1_9_LSB

```
double AF1_9_LSB = TWO_N38  [constexpr]
```
Definition at line 216 of file Galileo_INAV.h.

10.2.6.6.19 AF2_4_LSB

```
double AF2_4_LSB = TWO_N59  [constexpr]
```
Definition at line 113 of file Galileo_INAV.h.

10.2.6.6.20 AI0_5_LSB

```
double AI0_5_LSB = TWO_N2  [constexpr]
```
Definition at line 121 of file Galileo_INAV.h.

10.2.6.6.21 AI1_5_LSB

```
double AI1_5_LSB = TWO_N8  [constexpr]
```
Definition at line 123 of file Galileo_INAV.h.

10.2.6.6.22 AI2_5_LSB

```
double AI2_5_LSB = TWO_N15  [constexpr]
```
Definition at line 125 of file Galileo_INAV.h.

10.2.6.6.23 ALM_AF0_LSB

```
double ALM_AF0_LSB = TWO_N20  [constexpr]
```
Definition at line 292 of file GPS_L1_CA.h.

10.2.6.6.24 ALM_AF1_LSB

```
double ALM_AF1_LSB = TWO_N38  [constexpr]
```
Definition at line 294 of file GPS_L1_CA.h.

10.2.6.6.25 ALM_DELTAI_LSB

```
double ALM_DELTAI_LSB = TWO_N19  [constexpr]
```
Definition at line 278 of file GPS_L1_CA.h.

10.2.6.6.26 ALM_ECC_LSB

```
double ALM_ECC_LSB = TWO_N21  [constexpr]
```
Definition at line 274 of file GPS_L1_CA.h.

10.2.6.6.27 ALM_MZERO_LSB

```
double ALM_MZERO_LSB = TWO_N23  [constexpr]
```
Definition at line 289 of file GPS_L1_CA.h.

10.2.6.6.28 ALM_OMEGA_LSB

```
double ALM_OMEGA_LSB = TWO_N23  [constexpr]
```
Definition at line 287 of file GPS_L1_CA.h.

10.2.6.6.29 ALM_OMEGADOT_LSB

```
double ALM_OMEGADOT_LSB = TWO_N38  [constexpr]
```
Definition at line 280 of file GPS_L1_CA.h.

10.2.6.6.30 ALM_OMEGAZERO_LSB

```
double ALM_OMEGAZERO_LSB = TWO_N23  [constexpr]
```
Definition at line 285 of file GPS_L1_CA.h.

10.2.6.6.31 ALM_SQUAREA_LSB

```
double ALM_SQUAREA_LSB = TWO_N11  [constexpr]
```
Definition at line 283 of file GPS_L1_CA.h.

10.2.6.6.32 ALM_TOA_LSB

```
int32_t ALM_TOA_LSB = static_cast<int32_t>(TWO_P12)  [constexpr]
```
Definition at line 276 of file GPS_L1_CA.h.

10.2.6.6.33 ALPHA_0_LSB

```
double ALPHA_0_LSB = TWO_N30  [constexpr]
```
Definition at line 154 of file GPS_L1_CA.h.

10.2.6.6.34 ALPHA_1_LSB

```
double ALPHA_1_LSB = TWO_N27  [constexpr]
```
Definition at line 156 of file GPS_L1_CA.h.

10.2.6.6.35 ALPHA_2_LSB

```
double ALPHA_2_LSB = TWO_N24  [constexpr]
```
Definition at line 158 of file GPS_L1_CA.h.

10.2.6.6.36 ALPHA_3_LSB

```
double ALPHA_3_LSB = TWO_N24  [constexpr]
```
Definition at line 160 of file GPS_L1_CA.h.

10.2.6.6.37 AODO_LSB

```
int32_t AODO_LSB = 900  [constexpr]
```
Definition at line 125 of file GPS_L1_CA.h.

10.2.6.6.38 AS2R

```
double AS2R = D2R / 3600.0  [constexpr]
```
arc sec to radian
Definition at line 126 of file MATH_CONSTANTS.h.

10.2.6.6.39 AU

```
double AU = 149597870691.0  [constexpr]
```
1 Astronomical Unit AU (m) distance from Earth to the Sun.
Definition at line 128 of file MATH_CONSTANTS.h.

10.2.6.6.40 BDS_EPH_SOURCE_CNAV1

```
int32_t BDS_EPH_SOURCE_CNAV1 = 7  [constexpr]
```
RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV1 (DNAV uses 0–6).
Definition at line 45 of file Beidou_CNAV1.h.

10.2.6.6.41 BDS_EPH_SOURCE_CNAV2

```
int32_t BDS_EPH_SOURCE_CNAV2 = 9  [constexpr]
```
RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV2.
Definition at line 57 of file Beidou_CNAV2.h.

10.2.6.6.42 BEIDOU_B1C_CODE_LENGTH_CHIPS

```
double BEIDOU_B1C_CODE_LENGTH_CHIPS = 10230.0  [constexpr]
```

§5.2.1 [chips]
Definition at line 31 of file Beidou_B1C.h.

10.2.6.6.43 BEIDOU_B1C_CODE_PERIOD_MS

```
uint32_t BEIDOU_B1C_CODE_PERIOD_MS = 10  [constexpr]
```

§5.2.1 [ms]
Definition at line 33 of file Beidou_B1C.h.

10.2.6.6.44 BEIDOU_B1C_CODE_PERIOD_S

```
double BEIDOU_B1C_CODE_PERIOD_S = 0.01  [constexpr]
```

§5.2.1, 10 ms [s]
Definition at line 32 of file Beidou_B1C.h.

10.2.6.6.45 BEIDOU_B1C_CODE_RATE_CPS

```
double BEIDOU_B1C_CODE_RATE_CPS = 1.023e6  [constexpr]
```

§5.2.1 [chips/s]
Definition at line 30 of file Beidou_B1C.h.

10.2.6.6.46 BEIDOU_B1C_FRAME_LENGTH_SYMBOLS

```
int32_t BEIDOU_B1C_FRAME_LENGTH_SYMBOLS = 1800  [constexpr]
```

§6.2 [symbols]
Definition at line 38 of file Beidou_B1C.h.

10.2.6.6.47 BEIDOU_B1C_FRAME_PERIOD_S

```
int32_t BEIDOU_B1C_FRAME_PERIOD_S = 18  [constexpr]
```

§6.2 [s]
Definition at line 39 of file Beidou_B1C.h.

10.2.6.6.48 BEIDOU_B1C_FREQ_HZ

```
double BEIDOU_B1C_FREQ_HZ = FREQ1  [constexpr]
```

§4.2.2, Table 4-1 [Hz]
Definition at line 29 of file Beidou_B1C.h.

10.2.6.6.49 BEIDOU_B1C_NUMBER_OF_PRNS

```
int32_t BEIDOU_B1C_NUMBER_OF_PRNS = 63  [constexpr]
```

§5.2.1
Definition at line 40 of file Beidou_B1C.h.

10.2.6.6.50 BEIDOU_B1C_OPT_ACQ_FS_SPS

```
uint32_t BEIDOU_B1C_OPT_ACQ_FS_SPS = 4000000  [constexpr]
```

4 samples/chip at 1.023 Mcps
Definition at line 34 of file Beidou_B1C.h.

10.2.6.6.51 BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH

```
int32_t BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH = 1800  [constexpr]
```

§5.2.2 [chips]
Definition at line 35 of file Beidou_B1C.h.

10.2.6.6.52 BEIDOU_B1C_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B1C_SYMBOLS_PER_BIT = 1  [constexpr]
```

100 sps, 10 ms/code period
Definition at line 37 of file Beidou_B1C.h.

10.2.6.6.53 BEIDOU_B1C_TELEMETRY_RATE_SPS

```
int32_t BEIDOU_B1C_TELEMETRY_RATE_SPS = 100  [constexpr]
```

§6.2 [symbols/s]
Definition at line 36 of file Beidou_B1C.h.

10.2.6.6.54 BEIDOU_B1I_CODE_LENGTH_CHIPS

```
double BEIDOU_B1I_CODE_LENGTH_CHIPS = 2046.0  [constexpr]
```

Beidou B1I code length [chips].
Definition at line 34 of file Beidou_B1I.h.

10.2.6.6.55 BEIDOU_B1I_CODE_PERIOD_MS

```
uint32_t BEIDOU_B1I_CODE_PERIOD_MS = 1  [constexpr]
```

Beidou B1I code period [ms].
Definition at line 38 of file Beidou_B1I.h.

10.2.6.6.56 BEIDOU_B1I_CODE_PERIOD_S

```
double BEIDOU_B1I_CODE_PERIOD_S = 0.001  [constexpr]
```

Beidou B1I code period [seconds].
Definition at line 35 of file Beidou_B1I.h.

10.2.6.6.57 BEIDOU_B1I_CODE_RATE_CPS

```
double BEIDOU_B1I_CODE_RATE_CPS = 2.046e6  [constexpr]
```

Beidou B1I code rate [chips/s].
Definition at line 33 of file Beidou_B1I.h.

10.2.6.6.58 BEIDOU_B1I_D2_SECONDARY_CODE_STR

```
char BEIDOU_B1I_D2_SECONDARY_CODE_STR[3] = "00"  [constexpr]
```

Definition at line 51 of file Beidou_B1I.h.

10.2.6.6.59 BEIDOU_B1I_FREQ_HZ

```
double BEIDOU_B1I_FREQ_HZ = FREQ1_BDS  [constexpr]
```

B1I [Hz].
Definition at line 32 of file Beidou_B1I.h.

10.2.6.6.60 BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS

```
int32_t BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22  [constexpr]
```

Definition at line 42 of file Beidou_B1I.h.

10.2.6.6.61 BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR

```
char BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR[23] = "111111000001100001100"  [constexpr]
```

Definition at line 50 of file Beidou_B1I.h.

10.2.6.6.62 BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2  [constexpr]
```

Definition at line 46 of file Beidou_B1I.h.

10.2.6.6.63 BEIDOU_B1I_OPT_ACQ_FS_SPS

```
double BEIDOU_B1I_OPT_ACQ_FS_SPS = 4e6  [constexpr]
```

∼2 samples/chip (same ratio as GPS L1 at 2 MHz)
Definition at line 37 of file Beidou_B1I.h.

10.2.6.6.64 BEIDOU_B1I_PREAMBLE_DURATION_MS

```
int32_t BEIDOU_B1I_PREAMBLE_DURATION_MS = 220  [constexpr]
```

Definition at line 43 of file Beidou_B1I.h.

10.2.6.6.65 BEIDOU_B1I_PREAMBLE_DURATION_S

```
double BEIDOU_B1I_PREAMBLE_DURATION_S = 0.220  [constexpr]
```

Definition at line 36 of file Beidou_B1I.h.

10.2.6.6.66 BEIDOU_B1I_PREAMBLE_LENGTH_BITS

```
uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_BITS = 11  [constexpr]
```

Definition at line 39 of file Beidou_B1I.h.

10.2.6.6.67 BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS

```
uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS = 220  [constexpr]
```

Definition at line 40 of file Beidou_B1I.h.

10.2.6.6.68 BEIDOU_B1I_SECONDARY_CODE_LENGTH

```
int32_t BEIDOU_B1I_SECONDARY_CODE_LENGTH = 20  [constexpr]
```

Definition at line 41 of file Beidou_B1I.h.

10.2.6.6.69 BEIDOU_B1I_SECONDARY_CODE_STR

```
char BEIDOU_B1I_SECONDARY_CODE_STR[21] = "00000100110101001110"  [constexpr]
```

Definition at line 49 of file Beidou_B1I.h.

10.2.6.6.70 BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND

```
int32_t BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND = 50  [constexpr]
```

Definition at line 44 of file Beidou_B1I.h.

10.2.6.6.71 BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND

```
int32_t BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND *
BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT  [constexpr]
```

Definition at line 48 of file Beidou_B1I.h.

10.2.6.6.72 BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS

```
int32_t BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_cast<uint32_↩
t>(BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT) * BEIDOU_B1I_CODE_PERIOD_MS)  [constexpr]
```

Definition at line 47 of file Beidou_B1I.h.

10.2.6.6.73 BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT = 20  [constexpr]
```

Definition at line 45 of file Beidou_B1I.h.

10.2.6.6.74 BEIDOU_B2A_CODE_LENGTH_CHIPS

```
double BEIDOU_B2A_CODE_LENGTH_CHIPS = 10230.0  [constexpr]
```

B2a primary code length [chips].
Definition at line 30 of file Beidou_B2a.h.

10.2.6.6.75 BEIDOU_B2A_CODE_PERIOD_MS

```
uint32_t BEIDOU_B2A_CODE_PERIOD_MS = 1  [constexpr]
```

Definition at line 33 of file Beidou_B2a.h.

10.2.6.6.76 BEIDOU_B2A_CODE_PERIOD_S

```
double BEIDOU_B2A_CODE_PERIOD_S = 0.001  [constexpr]
```

B2a primary code period [seconds].
Definition at line 31 of file Beidou_B2a.h.

10.2.6.6.77 BEIDOU_B2A_CODE_RATE_CPS

```
double BEIDOU_B2A_CODE_RATE_CPS = 10.23e6  [constexpr]
```

B2a primary code rate [chips/s].
Definition at line 29 of file Beidou_B2a.h.

10.2.6.6.78 BEIDOU_B2A_FREQ_HZ

```
double BEIDOU_B2A_FREQ_HZ = FREQ5  [constexpr]
```

BeiDou B2a [Hz].
Definition at line 28 of file Beidou_B2a.h.

10.2.6.6.79 BEIDOU_B2A_OPT_ACQ_FS_SPS

```
double BEIDOU_B2A_OPT_ACQ_FS_SPS = 20e6  [constexpr]
```

Definition at line 32 of file Beidou_B2a.h.

10.2.6.6.80 BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS

```
uint32_t BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS = 24  [constexpr]
```

Definition at line 34 of file Beidou_B2a.h.

10.2.6.6.81 BEIDOU_B2A_PREAMBLE_SYMBOLS_STR

```
char BEIDOU_B2A_PREAMBLE_SYMBOLS_STR[25] = "111000100100110111101000"  [constexpr]
```

0xE24DE8
Definition at line 39 of file Beidou_B2a.h.

10.2.6.6.82 BEIDOU_B2A_SECONDARY_CODE_LENGTH

```
int32_t BEIDOU_B2A_SECONDARY_CODE_LENGTH = 5  [constexpr]
```

Definition at line 37 of file Beidou_B2a.h.

10.2.6.6.83 BEIDOU_B2A_SECONDARY_CODE_STR

```
char BEIDOU_B2A_SECONDARY_CODE_STR[6] = "00010"  [constexpr]
```

ICD data secondary code, MSB first.
Definition at line 38 of file Beidou_B2a.h.

10.2.6.6.84 BEIDOU_B2A_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B2A_SYMBOLS_PER_BIT = 1  [constexpr]
```

Definition at line 35 of file Beidou_B2a.h.

10.2.6.6.85 BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND

```
int32_t BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND = 1000  [constexpr]
```
Definition at line 36 of file Beidou_B2a.h.

10.2.6.6.86 BEIDOU_B3I_CODE_LENGTH_CHIPS

```
double BEIDOU_B3I_CODE_LENGTH_CHIPS = 10230.0  [constexpr]
```
BeiDou B3I code length [chips].
Definition at line 33 of file Beidou_B3I.h.

10.2.6.6.87 BEIDOU_B3I_CODE_PERIOD_MS

```
uint32_t BEIDOU_B3I_CODE_PERIOD_MS = 1  [constexpr]
```
BeiDou B3I code period [ms].
Definition at line 37 of file Beidou_B3I.h.

10.2.6.6.88 BEIDOU_B3I_CODE_PERIOD_S

```
double BEIDOU_B3I_CODE_PERIOD_S = 0.001  [constexpr]
```
BeiDou B3I code period [seconds].
Definition at line 34 of file Beidou_B3I.h.

10.2.6.6.89 BEIDOU_B3I_CODE_RATE_CPS

```
double BEIDOU_B3I_CODE_RATE_CPS = 10.23e6  [constexpr]
```
BeiDou B3I code rate [chips/s].
Definition at line 32 of file Beidou_B3I.h.

10.2.6.6.90 BEIDOU_B3I_D2_SECONDARY_CODE_STR

```
char BEIDOU_B3I_D2_SECONDARY_CODE_STR[3] = "00"  [constexpr]
```
Definition at line 50 of file Beidou_B3I.h.

10.2.6.6.91 BEIDOU_B3I_FREQ_HZ

```
double BEIDOU_B3I_FREQ_HZ = FREQ3_BDS  [constexpr]
```
BeiDou B3I [Hz].
Definition at line 31 of file Beidou_B3I.h.

10.2.6.6.92 BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS

```
int32_t BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22  [constexpr]
```
Definition at line 41 of file Beidou_B3I.h.

10.2.6.6.93 BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR

```
char BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR[23] = "111111000000110000 1100"  [constexpr]
```
Definition at line 49 of file Beidou_B3I.h.

10.2.6.6.94 BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2  [constexpr]
```
Definition at line 45 of file Beidou_B3I.h.

10.2.6.6.95 BEIDOU_B3I_OPT_ACQ_FS_SPS

```
double BEIDOU_B3I_OPT_ACQ_FS_SPS = 10e6  [constexpr]
```
Definition at line 36 of file Beidou_B3I.h.

10.2.6.6.96 BEIDOU_B3I_PREAMBLE_DURATION_MS

```
int32_t BEIDOU_B3I_PREAMBLE_DURATION_MS = 220  [constexpr]
```
Definition at line 42 of file Beidou_B3I.h.

10.2.6.6.97 BEIDOU_B3I_PREAMBLE_DURATION_S

```
double BEIDOU_B3I_PREAMBLE_DURATION_S = 0.220  [constexpr]
```
Definition at line 35 of file Beidou_B3I.h.

10.2.6.6.98 BEIDOU_B3I_PREAMBLE_LENGTH_BITS

```
uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_BITS = 11  [constexpr]
```
Definition at line 38 of file Beidou_B3I.h.

10.2.6.6.99 BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS

```
uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS = 220  [constexpr]
```
Definition at line 39 of file Beidou_B3I.h.

10.2.6.6.100 BEIDOU_B3I_SECONDARY_CODE_LENGTH

```
int32_t BEIDOU_B3I_SECONDARY_CODE_LENGTH = 20  [constexpr]
```
Definition at line 40 of file Beidou_B3I.h.

10.2.6.6.101 BEIDOU_B3I_SECONDARY_CODE_STR

```
char BEIDOU_B3I_SECONDARY_CODE_STR[21] = "00000100110101001110"  [constexpr]
```
Definition at line 48 of file Beidou_B3I.h.

10.2.6.6.102 BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND

```
int32_t BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND = 50  [constexpr]
```
D1 NAV message bit rate [bits/s].
Definition at line 43 of file Beidou_B3I.h.

10.2.6.6.103 BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND

```
int32_t BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND *
BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT  [constexpr]
```
Definition at line 47 of file Beidou_B3I.h.

10.2.6.6.104 BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS

```
int32_t BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_cast<uint32_↩
t>(BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT) * BEIDOU_B3I_CODE_PERIOD_MS)  [constexpr]
```
Definition at line 46 of file Beidou_B3I.h.

10.2.6.6.105 BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT = 20  [constexpr]
```
Definition at line 44 of file Beidou_B3I.h.

10.2.6.6.106 BEIDOU_CNAV1_A_DOT_LSB

```
double BEIDOU_CNAV1_A_DOT_LSB = TWO_N21  [constexpr]
```
2^-21 m/s, Table 7-8
Definition at line 51 of file Beidou_CNAV1.h.

10.2.6.6.107 BEIDOU_CNAV1_A_REF_IGSO

```
double BEIDOU_CNAV1_A_REF_IGSO = 42162200.0  [constexpr]
```
Definition at line 48 of file Beidou_CNAV1.h.

10.2.6.6.108 BEIDOU_CNAV1_A_REF_MEO

```
double BEIDOU_CNAV1_A_REF_MEO = 27906100.0  [constexpr]
```
Definition at line 47 of file Beidou_CNAV1.h.

10.2.6.6.109 BEIDOU_CNAV1_AF0_LSB

```
double BEIDOU_CNAV1_AF0_LSB = TWO_N34  [constexpr]
```
Definition at line 66 of file Beidou_CNAV1.h.

10.2.6.6.110 BEIDOU_CNAV1_AF1_LSB

```
double BEIDOU_CNAV1_AF1_LSB = TWO_N50  [constexpr]
```
Definition at line 67 of file Beidou_CNAV1.h.

10.2.6.6.111 BEIDOU_CNAV1_AF2_LSB

```
double BEIDOU_CNAV1_AF2_LSB = TWO_N66  [constexpr]
```
Definition at line 68 of file Beidou_CNAV1.h.

10.2.6.6.112 BEIDOU_CNAV1_CIC_LSB

```
double BEIDOU_CNAV1_CIC_LSB = TWO_N30  [constexpr]
```
Definition at line 61 of file Beidou_CNAV1.h.

10.2.6.6.113 BEIDOU_CNAV1_CIS_LSB

```
double BEIDOU_CNAV1_CIS_LSB = TWO_N30  [constexpr]
```
Definition at line 60 of file Beidou_CNAV1.h.

10.2.6.6.114 BEIDOU_CNAV1_CRC_BITS

```
int32_t BEIDOU_CNAV1_CRC_BITS = 24  [constexpr]
```
Definition at line 40 of file Beidou_CNAV1.h.

10.2.6.6.115 BEIDOU_CNAV1_CRC_LSB

```
double BEIDOU_CNAV1_CRC_LSB = TWO_N8  [constexpr]
```
Definition at line 63 of file Beidou_CNAV1.h.

10.2.6.6.116 BEIDOU_CNAV1_CRS_LSB

```
double BEIDOU_CNAV1_CRS_LSB = TWO_N8  [constexpr]
```
Definition at line 62 of file Beidou_CNAV1.h.

10.2.6.6.117 BEIDOU_CNAV1_CUC_LSB

```
double BEIDOU_CNAV1_CUC_LSB = TWO_N30  [constexpr]
```
Definition at line 65 of file Beidou_CNAV1.h.

10.2.6.6.118 BEIDOU_CNAV1_CUS_LSB

```
double BEIDOU_CNAV1_CUS_LSB = TWO_N30  [constexpr]
```
Definition at line 64 of file Beidou_CNAV1.h.

10.2.6.6.119 BEIDOU_CNAV1_DELTA_A_LSB

```
double BEIDOU_CNAV1_DELTA_A_LSB = TWO_N9  [constexpr]
```
Definition at line 50 of file Beidou_CNAV1.h.

10.2.6.6.120 BEIDOU_CNAV1_DELTA_N0_DOT_LSB

```
double BEIDOU_CNAV1_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI  [constexpr]
```
Definition at line 53 of file Beidou_CNAV1.h.

10.2.6.6.121 BEIDOU_CNAV1_DELTA_N0_LSB

```
double BEIDOU_CNAV1_DELTA_N0_LSB = TWO_N44 * GNSS_PI  [constexpr]
```
Definition at line 52 of file Beidou_CNAV1.h.

10.2.6.6.122 BEIDOU_CNAV1_E_LSB

```
double BEIDOU_CNAV1_E_LSB = TWO_N34  [constexpr]
```
Definition at line 55 of file Beidou_CNAV1.h.

10.2.6.6.123 BEIDOU_CNAV1_FRAME_PERIOD_S

```
int32_t BEIDOU_CNAV1_FRAME_PERIOD_S = 18  [constexpr]
```
Definition at line 29 of file Beidou_CNAV1.h.

10.2.6.6.124 BEIDOU_CNAV1_FRAME_SYMBOLS

```
int32_t BEIDOU_CNAV1_FRAME_SYMBOLS = 1800  [constexpr]
```
Definition at line 28 of file Beidou_CNAV1.h.

10.2.6.6.125 BEIDOU_CNAV1_H100_200_ELEMENT

```
uint8_t BEIDOU_CNAV1_H100_200_ELEMENT[400]  [constexpr]
```
Initial value:

```
= {
35, 13, 51, 60, 1, 44, 53, 24, 1, 45, 15, 6, 45, 15, 6, 1, 1, 44, 53, 24, 1,
45, 15, 6, 35, 46, 56, 15, 6, 1, 45, 15, 15, 6, 1, 45, 44, 53, 24, 1, 24, 1,
44, 30, 1, 45, 15, 6, 30, 24, 1, 44, 24, 1, 44, 30, 45, 15, 6, 1, 17, 38, 49,
11, 24, 1, 44, 30, 24, 1, 44, 53, 24, 1, 44, 53, 30, 24, 1, 44, 33, 42, 14,
24, 33, 42, 14, 24, 45, 15, 6, 1, 1, 45, 15, 6, 30, 24, 1, 44, 24, 1, 44, 53,
1, 44, 30, 24, 57, 25, 9, 41, 1, 45, 15, 6, 1, 45, 15, 6, 42, 36, 12, 57, 6,
1, 45, 15, 24, 1, 44, 53, 24, 1, 44, 30, 1, 45, 15, 6, 1, 45, 15, 6, 44, 53,
24, 1, 30, 24, 1, 44, 1, 44, 30, 24, 53, 24, 1, 44, 1, 44, 53, 24, 27, 28,
30, 31, 53, 24, 1, 44, 24, 1, 44, 30, 45, 15, 6, 1, 30, 24, 1, 44, 1, 45, 15,
6, 26, 22, 14, 2, 35, 13, 18, 60, 45, 15, 6, 1, 30, 1, 44, 7, 6, 1, 45, 15,
6, 1, 45, 15, 53, 24, 1, 44, 24, 1, 44, 53, 30, 24, 1, 44, 1, 44, 30, 24, 44,
53, 24, 1, 53, 24, 1, 44, 44, 30, 24, 1, 30, 24, 1, 44, 1, 44, 30, 24, 1, 44,
30, 24, 41, 16, 29, 51, 1, 44, 30, 24, 38, 23, 22, 7, 44, 53, 24, 1, 1, 45,
15, 6, 30, 24, 1, 44, 53, 24, 1, 44, 6, 1, 45, 15, 24, 1, 44, 53, 35, 46, 56,
15, 5, 33, 42, 14, 54, 7, 38, 23, 1, 45, 15, 6, 44, 30, 24, 1, 6, 1, 45, 15,
53, 24, 1, 44, 44, 53, 24, 1, 1, 44, 53, 24, 1, 44, 30, 24, 44, 30, 24, 1, 1,
44, 53, 24, 45, 15, 6, 1, 6, 1, 45, 15, 1, 44, 53, 24, 42, 47, 37, 32, 51, 60,
35, 13, 29, 28, 30, 31, 6, 1, 45, 15, 24, 1, 44, 53, 44, 53, 24, 1, 44, 30,
24, 1, 38, 49, 11, 17, 44, 30, 24, 1, 24, 1, 44, 30, 24, 1, 44, 30, 1, 44, 53,
24, 53, 24, 1, 44}
```
Definition at line 69 of file beidou_cnav1_ldpc.h.

10.2.6.6.126 BEIDOU_CNAV1_H100_200_INDEX

```
uint16_t BEIDOU_CNAV1_H100_200_INDEX[400]  [constexpr]
```
Initial value:

```
= {
11, 62, 102, 150, 9, 60, 100, 148, 0, 51, 142, 197, 22, 80, 116, 154, 4, 90,
131, 177, 47, 95, 138, 191, 51, 79, 146, 195, 44, 75, 142, 190, 13, 57, 135,
198, 24, 65, 120, 173, 6, 88, 129, 179, 7, 89, 130, 176, 6, 58, 106, 158, 8,
60, 108, 160, 44, 92, 139, 188, 4, 56, 104, 156, 10, 61, 101, 149, 39, 87, 123,
168, 15, 67, 105, 167, 50, 78, 145, 194, 17, 98, 151, 187, 46, 94, 137, 190,
14, 66, 104, 166, 7, 59, 107, 159, 21, 83, 119, 153, 31, 87, 114, 167, 2, 49,
140, 199, 12, 64, 106, 164, 40, 53, 132, 159, 19, 96, 149, 185, 16, 68, 112,
```

```
168, 14, 58, 132, 199, 34, 69, 125, 162, 23, 75, 119, 175, 42, 96, 144, 192,
8, 63, 103, 151, 23, 81, 117, 155, 24, 93, 111, 182, 20, 72, 116, 172, 17, 69,
113, 169, 34, 82, 130, 182, 1, 53, 101, 153, 46, 73, 140, 188, 13, 65, 107,
165, 2, 54, 102, 154, 18, 70, 114, 170, 26, 67, 122, 175, 29, 77, 125, 177,
36, 84, 120, 169, 25, 94, 108, 183, 39, 89, 137, 185, 21, 73, 117, 173, 28,
76, 124, 176, 36, 90, 138, 186, 33, 68, 124, 161, 12, 56, 134, 197, 29, 85,
112, 165, 45, 93, 136, 189, 27, 64, 123, 172, 28, 84, 115, 164, 25, 66, 121,
174, 37, 85, 121, 170, 3, 50, 141, 196, 48, 76, 147, 192, 35, 70, 126, 163,
32, 80, 128, 180, 0, 52, 100, 152, 43, 52, 135, 158, 35, 83, 131, 183, 10,
62, 110, 162, 19, 71, 115, 171, 15, 59, 133, 196, 33, 81, 129, 181, 41, 54,
133, 156, 20, 82, 118, 152, 38, 86, 122, 171, 30, 78, 126, 178, 9, 61, 109,
161, 26, 95, 109, 180, 45, 72, 143, 191, 1, 48, 143, 198, 40, 98, 146, 194,
18, 99, 148, 184, 5, 57, 105, 157, 41, 99, 147, 195, 31, 79, 127, 179, 3, 55,
103, 155, 22, 74, 118, 174, 37, 91, 139, 187, 5, 91, 128, 178, 30, 86, 113,
166, 43, 97, 145, 193, 16, 97, 150, 186, 11, 63, 111, 163, 32, 71, 127, 160,
42, 55, 134, 157, 38, 88, 136, 184, 47, 74, 141, 189, 49, 77, 144, 193, 27,
92, 110, 181}
```

Definition at line 42 of file beidou_cnav1_ldpc.h.

10.2.6.6.127 BEIDOU_CNAV1_H44_88_ELEMENT

```
uint8_t BEIDOU_CNAV1_H44_88_ELEMENT[176]  [constexpr]
```

Initial value:

```
                                        = {
30, 24, 1, 44, 24, 1, 44, 30, 40, 32, 61, 18, 53, 24, 1, 44, 51, 60, 35, 13, 18,
15, 32, 61, 15, 6, 1, 45, 30, 24, 1, 44, 6, 1, 45, 15, 45, 15, 6, 1, 1, 45, 15,
6, 1, 44, 53, 24, 24, 1, 44, 53, 44, 30, 24, 1, 34, 33, 45, 36, 55, 9, 34, 3,
1, 44, 53, 24, 61, 47, 20, 8, 53, 24, 1, 44, 15, 6, 1, 45, 13, 18, 60, 35, 45,
15, 6, 1, 24, 1, 44, 53, 37, 32, 52, 47, 44, 53, 24, 1, 39, 36, 34, 33, 44, 35,
31, 50, 12, 25, 36, 14, 15, 35, 46, 56, 53, 24, 1, 44, 1, 44, 53, 24, 24, 1, 44,
30, 44, 30, 24, 1, 15, 6, 1, 45, 30, 24, 1, 44, 2, 50, 22, 14, 33, 42, 14, 5, 34,
3, 55, 9, 44, 35, 61, 50, 15, 6, 1, 45, 45, 15, 6, 1, 1, 44, 30, 24, 6, 1, 45,
15, 1, 44, 53, 24}
```

Definition at line 104 of file beidou_cnav1_ldpc.h.

10.2.6.6.128 BEIDOU_CNAV1_H44_88_INDEX

```
uint16_t BEIDOU_CNAV1_H44_88_INDEX[176]  [constexpr]
```

Initial value:

```
                                        = {
14, 35, 56, 70, 11, 29, 55, 73, 13, 39, 53, 69, 15, 34, 57, 71, 1, 27, 45, 54,
23, 41, 63, 87, 2, 20, 46, 68, 6, 24, 50, 61, 2, 26, 61, 79, 9, 33, 59, 77, 4,
30, 48, 74, 22, 42, 59, 76, 12, 38, 52, 68, 23, 43, 58, 77, 19, 21, 63, 64, 11,
25, 65, 82, 17, 39, 44, 75, 9, 35, 49, 72, 19, 29, 66, 84, 13, 36, 56, 82, 17,
43, 67, 81, 22, 40, 62, 86, 3, 21, 47, 69, 10, 24, 64, 83, 0, 37, 70, 86, 5, 31,
49, 75, 4, 40, 53, 84, 5, 41, 52, 85, 18, 28, 67, 85, 0, 26, 44, 55, 10, 28, 54,
72, 7, 30, 50, 81, 1, 36, 71, 87, 16, 38, 45, 74, 8, 34, 48, 73, 8, 32, 58, 76,
12, 37, 57, 83, 6, 31, 51, 80, 15, 33, 47, 79, 16, 42, 66, 80, 7, 25, 51, 60, 3,
27, 60, 78, 14, 32, 46, 78, 18, 20, 62, 65}
```

Definition at line 92 of file beidou_cnav1_ldpc.h.

10.2.6.6.129 BEIDOU_CNAV1_I0_LSB

```
double BEIDOU_CNAV1_I0_LSB = TWO_N32 * GNSS_PI  [constexpr]
```

Definition at line 57 of file Beidou_CNAV1.h.

10.2.6.6.130 BEIDOU_CNAV1_IDOT_LSB

```
double BEIDOU_CNAV1_IDOT_LSB = TWO_N44 * GNSS_PI  [constexpr]
```

Definition at line 59 of file Beidou_CNAV1.h.

10.2.6.6.131 BEIDOU_CNAV1_INTERLEAVE_COLS

```
int32_t BEIDOU_CNAV1_INTERLEAVE_COLS = 48  [constexpr]
```

Definition at line 37 of file Beidou_CNAV1.h.

10.2.6.6.132 BEIDOU_CNAV1_INTERLEAVE_ROWS

```
int32_t BEIDOU_CNAV1_INTERLEAVE_ROWS = 36  [constexpr]
```

Definition at line 36 of file Beidou_CNAV1.h.

10.2.6.6.133 BEIDOU_CNAV1_INTERLEAVED_SYMBOLS

```
int32_t BEIDOU_CNAV1_INTERLEAVED_SYMBOLS = 1728  [constexpr]
```
Definition at line 35 of file Beidou_CNAV1.h.

10.2.6.6.134 BEIDOU_CNAV1_ISC_LSB

```
double BEIDOU_CNAV1_ISC_LSB = TWO_N34  [constexpr]
```
Definition at line 70 of file Beidou_CNAV1.h.

10.2.6.6.135 BEIDOU_CNAV1_LDPC_DC200

```
int32_t BEIDOU_CNAV1_LDPC_DC200 = 4  [constexpr]
```
Definition at line 30 of file beidou_cnav1_ldpc.h.

10.2.6.6.136 BEIDOU_CNAV1_LDPC_DC88

```
int32_t BEIDOU_CNAV1_LDPC_DC88 = 4  [constexpr]
```
Definition at line 34 of file beidou_cnav1_ldpc.h.

10.2.6.6.137 BEIDOU_CNAV1_LDPC_K200

```
int32_t BEIDOU_CNAV1_LDPC_K200 = 100  [constexpr]
```
Definition at line 28 of file beidou_cnav1_ldpc.h.

10.2.6.6.138 BEIDOU_CNAV1_LDPC_K88

```
int32_t BEIDOU_CNAV1_LDPC_K88 = 44  [constexpr]
```
Definition at line 32 of file beidou_cnav1_ldpc.h.

10.2.6.6.139 BEIDOU_CNAV1_LDPC_M200

```
int32_t BEIDOU_CNAV1_LDPC_M200 = 100  [constexpr]
```
Definition at line 29 of file beidou_cnav1_ldpc.h.

10.2.6.6.140 BEIDOU_CNAV1_LDPC_M88

```
int32_t BEIDOU_CNAV1_LDPC_M88 = 44  [constexpr]
```
Definition at line 33 of file beidou_cnav1_ldpc.h.

10.2.6.6.141 BEIDOU_CNAV1_LDPC_MAX_ITER

```
int32_t BEIDOU_CNAV1_LDPC_MAX_ITER = BEIDOU_LDPC_MAX_ITER  [constexpr]
```
Definition at line 36 of file beidou_cnav1_ldpc.h.

10.2.6.6.142 BEIDOU_CNAV1_LDPC_N200

```
int32_t BEIDOU_CNAV1_LDPC_N200 = 200  [constexpr]
```
Definition at line 27 of file beidou_cnav1_ldpc.h.

10.2.6.6.143 BEIDOU_CNAV1_LDPC_N88

```
int32_t BEIDOU_CNAV1_LDPC_N88 = 88  [constexpr]
```
Definition at line 31 of file beidou_cnav1_ldpc.h.

10.2.6.6.144 BEIDOU_CNAV1_LDPC_NM

```
int32_t BEIDOU_CNAV1_LDPC_NM = BEIDOU_LDPC_NM  [constexpr]
```
Definition at line 35 of file beidou_cnav1_ldpc.h.

10.2.6.6.145 BEIDOU_CNAV1_M0_LSB

```
double BEIDOU_CNAV1_M0_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 54 of file Beidou_CNAV1.h.

10.2.6.6.146 BEIDOU_CNAV1_OMEGA_LSB

```
double BEIDOU_CNAV1_OMEGA_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 56 of file Beidou_CNAV1.h.

10.2.6.6.147 BEIDOU_CNAV1_OMEGADOT_LSB

```
double BEIDOU_CNAV1_OMEGADOT_LSB = TWO_N44 * GNSS_PI  [constexpr]
```
Definition at line 58 of file Beidou_CNAV1.h.

10.2.6.6.148 BEIDOU_CNAV1_PAGE_IONO_UTC

```
int32_t BEIDOU_CNAV1_PAGE_IONO_UTC = 1  [constexpr]
```
Definition at line 42 of file Beidou_CNAV1.h.

10.2.6.6.149 BEIDOU_CNAV1_SF2_DATA_BITS

```
int32_t BEIDOU_CNAV1_SF2_DATA_BITS = 600  [constexpr]
```
Definition at line 38 of file Beidou_CNAV1.h.

10.2.6.6.150 BEIDOU_CNAV1_SF3_DATA_BITS

```
int32_t BEIDOU_CNAV1_SF3_DATA_BITS = 264  [constexpr]
```
Definition at line 39 of file Beidou_CNAV1.h.

10.2.6.6.151 BEIDOU_CNAV1_SOH_LSB_S

```
int32_t BEIDOU_CNAV1_SOH_LSB_S = 18  [constexpr]
```
Definition at line 34 of file Beidou_CNAV1.h.

10.2.6.6.152 BEIDOU_CNAV1_SUBFRAME1_SYMBOLS

```
int32_t BEIDOU_CNAV1_SUBFRAME1_SYMBOLS = 72  [constexpr]
```
Definition at line 31 of file Beidou_CNAV1.h.

10.2.6.6.153 BEIDOU_CNAV1_SUBFRAME2_SYMBOLS

```
int32_t BEIDOU_CNAV1_SUBFRAME2_SYMBOLS = 1200  [constexpr]
```
Definition at line 32 of file Beidou_CNAV1.h.

10.2.6.6.154 BEIDOU_CNAV1_SUBFRAME3_SYMBOLS

```
int32_t BEIDOU_CNAV1_SUBFRAME3_SYMBOLS = 528  [constexpr]
```
Definition at line 33 of file Beidou_CNAV1.h.

10.2.6.6.155 BEIDOU_CNAV1_SYMBOL_RATE_SPS

```
int32_t BEIDOU_CNAV1_SYMBOL_RATE_SPS = 100  [constexpr]
```
Definition at line 30 of file Beidou_CNAV1.h.

10.2.6.6.156 BEIDOU_CNAV1_TGD_LSB

```
double BEIDOU_CNAV1_TGD_LSB = TWO_N34  [constexpr]
```
Definition at line 69 of file Beidou_CNAV1.h.

10.2.6.6.157 BEIDOU_CNAV1_TOE_TOC_LSB

```
double BEIDOU_CNAV1_TOE_TOC_LSB = 300.0  [constexpr]
```
Definition at line 49 of file Beidou_CNAV1.h.

10.2.6.6.158 BEIDOU_CNAV2_CRC_BITS

```
int32_t BEIDOU_CNAV2_CRC_BITS = 24  [constexpr]
```
Definition at line 37 of file Beidou_CNAV2.h.

10.2.6.6.159 BEIDOU_CNAV2_DATA_BITS

```
int32_t BEIDOU_CNAV2_DATA_BITS = BEIDOU_CNAV2_INFO_BITS - BEIDOU_CNAV2_CRC_BITS  [constexpr]
```
Definition at line 38 of file Beidou_CNAV2.h.

10.2.6.6.160 BEIDOU_CNAV2_FRAME_MS

```
int32_t BEIDOU_CNAV2_FRAME_MS = 3000  [constexpr]
```
Definition at line 32 of file Beidou_CNAV2.h.

10.2.6.6.161 BEIDOU_CNAV2_FRAME_PERIOD_S

```
int32_t BEIDOU_CNAV2_FRAME_PERIOD_S = 3  [constexpr]
```
Definition at line 29 of file Beidou_CNAV2.h.

10.2.6.6.162 BEIDOU_CNAV2_FRAME_SYMBOLS

```
int32_t BEIDOU_CNAV2_FRAME_SYMBOLS = 600  [constexpr]
```
Definition at line 28 of file Beidou_CNAV2.h.

10.2.6.6.163 BEIDOU_CNAV2_H48_96_ELEMENT

```
uint8_t BEIDOU_CNAV2_H48_96_ELEMENT[192]  [constexpr]
```
Initial value:

```
                                                = {
1, 45, 15, 6, 1, 44, 53, 24, 45, 15, 6, 1, 30, 24, 1, 44,
18, 15, 32, 61, 3, 55, 9, 34, 35, 31, 50, 44, 45, 15, 6, 1,
24, 1, 44, 53, 30, 24, 1, 44, 32, 42, 47, 37, 6, 1, 45, 15,
44, 53, 24, 1, 39, 36, 34, 33, 44, 53, 24, 1, 44, 53, 24, 1,
45, 15, 6, 1, 6, 1, 45, 15, 24, 1, 44, 53, 9, 41, 57, 58,
32, 61, 18, 40, 1, 45, 15, 6, 22, 14, 2, 50, 24, 1, 44, 30,
30, 24, 1, 44, 15, 46, 45, 44, 45, 15, 6, 1, 1, 44, 30, 24,
24, 1, 44, 53, 15, 6, 1, 45, 53, 24, 1, 44, 7, 38, 23, 54,
1, 45, 15, 6, 44, 53, 24, 1, 57, 25, 9, 41, 35, 13, 51, 60,
33, 45, 36, 34, 6, 1, 45, 15, 6, 1, 45, 15, 6, 1, 45, 15,
44, 35, 31, 50, 26, 27, 37, 5, 24, 1, 44, 30, 33, 42, 14, 5,
24, 1, 44, 30, 24, 1, 44, 30, 1, 44, 53, 24, 1, 44, 30, 24}
```
Definition at line 48 of file beidou_cnav2_ldpc.h.

10.2.6.6.164 BEIDOU_CNAV2_H48_96_INDEX

```
uint16_t BEIDOU_CNAV2_H48_96_INDEX[192]  [constexpr]
```
Initial value:

```
                                                = {
19, 46, 49, 76, 5, 29, 53, 71, 17, 30, 64, 72, 22, 36, 59, 82,
22, 41, 68, 94, 20, 44, 54, 75, 9, 41, 61, 86, 6, 47, 60, 89,
8, 40, 60, 87, 15, 26, 66, 81, 19, 24, 67, 95, 2, 26, 50, 72,
5, 38, 70, 89, 16, 34, 64, 92, 21, 45, 55, 74, 0, 24, 48, 78,
23, 37, 58, 83, 15, 43, 56, 91, 18, 47, 48, 77, 14, 42, 57, 90,
6, 30, 54, 76, 14, 27, 67, 80, 17, 35, 65, 93, 7, 46, 61, 88,
1, 25, 49, 79, 12, 45, 69, 79, 18, 25, 66, 94, 23, 40, 69, 95,
8, 36, 51, 84, 3, 38, 56, 86, 0, 29, 62, 85, 2, 39, 57, 87,
11, 33, 59, 81, 20, 43, 74, 93, 13, 32, 63, 91, 11, 35, 52, 83,
16, 31, 65, 73, 4, 28, 52, 70, 1, 28, 63, 84, 12, 33, 62, 90,
21, 42, 75, 92, 7, 31, 55, 77, 9, 37, 50, 85, 10, 34, 53, 82,
4, 39, 71, 88, 13, 44, 68, 78, 3, 27, 51, 73, 10, 32, 58, 80}
```
Definition at line 34 of file beidou_cnav2_ldpc.h.

10.2.6.6.165 BEIDOU_CNAV2_INFO_BITS

```
int32_t BEIDOU_CNAV2_INFO_BITS = 288  [constexpr]
```
Definition at line 36 of file Beidou_CNAV2.h.

10.2.6.6.166 BEIDOU_CNAV2_IODE_BITS

```
int32_t BEIDOU_CNAV2_IODE_BITS = 8  [constexpr]
```
Definition at line 54 of file Beidou_CNAV2.h.

10.2.6.6.167 BEIDOU_CNAV2_LDPC_DC

```
int32_t BEIDOU_CNAV2_LDPC_DC = 4  [constexpr]
```
Definition at line 30 of file beidou_cnav2_ldpc.h.

10.2.6.6.168 BEIDOU_CNAV2_LDPC_K

```
int32_t BEIDOU_CNAV2_LDPC_K = 48  [constexpr]
```
Definition at line 28 of file beidou_cnav2_ldpc.h.

10.2.6.6.169 BEIDOU_CNAV2_LDPC_M

```
int32_t BEIDOU_CNAV2_LDPC_M = 48  [constexpr]
```
Definition at line 29 of file beidou_cnav2_ldpc.h.

10.2.6.6.170 BEIDOU_CNAV2_LDPC_N

```
int32_t BEIDOU_CNAV2_LDPC_N = 96  [constexpr]
```
Definition at line 27 of file beidou_cnav2_ldpc.h.

10.2.6.6.171 BEIDOU_CNAV2_LDPC_SYMBOLS

```
int32_t BEIDOU_CNAV2_LDPC_SYMBOLS = 576  [constexpr]
```
encoded payload: 96 GF(64) symbols, 6 bits each
Definition at line 35 of file Beidou_CNAV2.h.

10.2.6.6.172 BEIDOU_CNAV2_MES_TYPE_BITS

```
int32_t BEIDOU_CNAV2_MES_TYPE_BITS = 6  [constexpr]
```
Definition at line 39 of file Beidou_CNAV2.h.

10.2.6.6.173 BEIDOU_CNAV2_MS_PER_SYMBOL

```
int32_t BEIDOU_CNAV2_MS_PER_SYMBOL = 5  [constexpr]
```
Definition at line 31 of file Beidou_CNAV2.h.

10.2.6.6.174 BEIDOU_CNAV2_MSG_ALM

```
int32_t BEIDOU_CNAV2_MSG_ALM = 40  [constexpr]
```
Definition at line 50 of file Beidou_CNAV2.h.

10.2.6.6.175 BEIDOU_CNAV2_MSG_CLK_ALM

```
int32_t BEIDOU_CNAV2_MSG_CLK_ALM = 31  [constexpr]
```
Definition at line 46 of file Beidou_CNAV2.h.

10.2.6.6.176 BEIDOU_CNAV2_MSG_CLK_DC

```
int32_t BEIDOU_CNAV2_MSG_CLK_DC = 34  [constexpr]
```
Definition at line 49 of file Beidou_CNAV2.h.

10.2.6.6.177 BEIDOU_CNAV2_MSG_CLK_EOP

```
int32_t BEIDOU_CNAV2_MSG_CLK_EOP = 32  [constexpr]
```
Definition at line 47 of file Beidou_CNAV2.h.

10.2.6.6.178 BEIDOU_CNAV2_MSG_CLK_IONO

```
int32_t BEIDOU_CNAV2_MSG_CLK_IONO = 30  [constexpr]
```
Definition at line 45 of file Beidou_CNAV2.h.

10.2.6.6.179 BEIDOU_CNAV2_MSG_CLK_UTC

```
int32_t BEIDOU_CNAV2_MSG_CLK_UTC = 33  [constexpr]
```
Definition at line 48 of file Beidou_CNAV2.h.

10.2.6.6.180 BEIDOU_CNAV2_MSG_EPH1

```
int32_t BEIDOU_CNAV2_MSG_EPH1 = 10  [constexpr]
```
Definition at line 43 of file Beidou_CNAV2.h.

10.2.6.6.181 BEIDOU_CNAV2_MSG_EPH2

```
int32_t BEIDOU_CNAV2_MSG_EPH2 = 11  [constexpr]
```
Definition at line 44 of file Beidou_CNAV2.h.

10.2.6.6.182 BEIDOU_CNAV2_MT10_IODE_BIT

```
int32_t BEIDOU_CNAV2_MT10_IODE_BIT = 53  [constexpr]
```
Definition at line 53 of file Beidou_CNAV2.h.

10.2.6.6.183 BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD

```
double BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD = 5.2  [constexpr]
```
on the normalized soft preamble statistic, about $|N(0,1)|$ on noise
Definition at line 41 of file Beidou_CNAV2.h.

10.2.6.6.184 BEIDOU_CNAV2_PREAMBLE_MS

```
int32_t BEIDOU_CNAV2_PREAMBLE_MS = 120  [constexpr]
```
24 symbols $*$ 5 ms secondary code
Definition at line 34 of file Beidou_CNAV2.h.

10.2.6.6.185 BEIDOU_CNAV2_PREAMBLE_SYMBOLS

```
int32_t BEIDOU_CNAV2_PREAMBLE_SYMBOLS = 24  [constexpr]
```
Definition at line 33 of file Beidou_CNAV2.h.

10.2.6.6.186 BEIDOU_CNAV2_SOW_LSB_S

```
int32_t BEIDOU_CNAV2_SOW_LSB_S = 3  [constexpr]
```
Definition at line 40 of file Beidou_CNAV2.h.

10.2.6.6.187 BEIDOU_CNAV2_SYMBOL_RATE_SPS

```
int32_t BEIDOU_CNAV2_SYMBOL_RATE_SPS = 200  [constexpr]
```
Definition at line 30 of file Beidou_CNAV2.h.

10.2.6.6.188 BEIDOU_DNAV_BCH_CODE_BITS

```
std::size_t BEIDOU_DNAV_BCH_CODE_BITS = 15  [constexpr]
```
Definition at line 87 of file Beidou_DNAV.h.

10.2.6.6.189 BEIDOU_DNAV_BCH_DATA_BITS

```
std::size_t BEIDOU_DNAV_BCH_DATA_BITS = 11  [constexpr]
```
Definition at line 88 of file Beidou_DNAV.h.

10.2.6.6.190 BEIDOU_DNAV_BCH_PARITY_BITS

```
std::size_t BEIDOU_DNAV_BCH_PARITY_BITS = 4  [constexpr]
```
Definition at line 89 of file Beidou_DNAV.h.

10.2.6.6.191 BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET

```
int32_t BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET = 14  [constexpr]
```
Definition at line 99 of file Beidou_DNAV.h.

10.2.6.6.192 BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET

```
int32_t BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET = 1356  [constexpr]
```
Definition at line 101 of file Beidou_DNAV.h.

10.2.6.6.193 BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS

```
double BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS = 6.0  [constexpr]
```
Definition at line 96 of file Beidou_DNAV.h.

10.2.6.6.194 BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS

```
double BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS = 3.0  [constexpr]
```
Definition at line 97 of file Beidou_DNAV.h.

10.2.6.6.195 BEIDOU_DNAV_PREAMBLE

```
char BEIDOU_DNAV_PREAMBLE[12] = "11100010010"  [constexpr]
```
Definition at line 106 of file Beidou_DNAV.h.

10.2.6.6.196 BEIDOU_DNAV_PREAMBLE_LENGTH_BITS

```
int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_BITS = 11  [constexpr]
```
Definition at line 91 of file Beidou_DNAV.h.

10.2.6.6.197 BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS

```
int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS = 11  [constexpr]
```
Definition at line 92 of file Beidou_DNAV.h.

10.2.6.6.198 BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS

```
int32_t BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS = 300  [constexpr]
```
Definition at line 93 of file Beidou_DNAV.h.

10.2.6.6.199 BEIDOU_DNAV_SECONDS_PER_WEEK

```
double BEIDOU_DNAV_SECONDS_PER_WEEK = 604800.0  [constexpr]
```
Definition at line 95 of file Beidou_DNAV.h.

10.2.6.6.200 BEIDOU_DNAV_SUBFRAME_DATA_BITS

```
int32_t BEIDOU_DNAV_SUBFRAME_DATA_BITS = 300  [constexpr]
```
Number of bits per subframe in the NAV message [bits].
Definition at line 94 of file Beidou_DNAV.h.

10.2.6.6.201 BEIDOU_DNAV_SUBFRAME_SYMBOLS

```
uint32_t BEIDOU_DNAV_SUBFRAME_SYMBOLS = 300  [constexpr]
```
Definition at line 103 of file Beidou_DNAV.h.

10.2.6.6.202 BEIDOU_DNAV_WORD_LENGTH_BITS

```
uint32_t BEIDOU_DNAV_WORD_LENGTH_BITS = 30  [constexpr]
```
Definition at line 105 of file Beidou_DNAV.h.

10.2.6.6.203 BEIDOU_DNAV_WORDS_SUBFRAME

```
uint32_t BEIDOU_DNAV_WORDS_SUBFRAME = 10  [constexpr]
```
Definition at line 104 of file Beidou_DNAV.h.

10.2.6.6.204 BEIDOU_F

```
double BEIDOU_F = -4.442807309e-10  [constexpr]
```
Constant, [s/(m)^(1/2)] F=-2(GM)^.5/C^2.
Definition at line 45 of file MATH_CONSTANTS.h.

10.2.6.6.205 BEIDOU_GM

```
double BEIDOU_GM = 3.986004418e14  [constexpr]
```
Universal gravitational constant times the mass of the Earth, [m^3/s^2] as defined in CGCS2000.
Definition at line 44 of file MATH_CONSTANTS.h.

10.2.6.6.206 BEIDOU_OMEGA_EARTH_DOT

```
double BEIDOU_OMEGA_EARTH_DOT = 7.2921150e-5  [constexpr]
```
Earth rotation rate, [rad/s] as defined in BDS-SIS-ICD-B1I-3.0 2019-02, pag. 3.
Definition at line 43 of file MATH_CONSTANTS.h.

10.2.6.6.207 BETA_0_LSB

```
double BETA_0_LSB = TWO_P11  [constexpr]
```
Definition at line 162 of file GPS_L1_CA.h.

10.2.6.6.208 BETA_1_LSB

```
double BETA_1_LSB = TWO_P14  [constexpr]
```
Definition at line 164 of file GPS_L1_CA.h.

10.2.6.6.209 BETA_2_LSB

```
double BETA_2_LSB = TWO_P16  [constexpr]
```
Definition at line 166 of file GPS_L1_CA.h.

10.2.6.6.210 BETA_3_LSB

```
double BETA_3_LSB = TWO_P16  [constexpr]
```
Definition at line 168 of file GPS_L1_CA.h.

10.2.6.6.211 BGD_E1_E5A_5_LSB

```
double BGD_E1_E5A_5_LSB = TWO_N32  [constexpr]
```
Definition at line 133 of file Galileo_INAV.h.

10.2.6.6.212 BGD_E1_E5B_5_LSB

```
double BGD_E1_E5B_5_LSB = TWO_N32  [constexpr]
```
Definition at line 135 of file Galileo_INAV.h.

10.2.6.6.213 BITS_IN_OCTET

```
int32_t BITS_IN_OCTET = 8  [constexpr]
```
Definition at line 276 of file Galileo_INAV.h.

10.2.6.6.214 C_IC_4_LSB

```
double C_IC_4_LSB = TWO_N29  [constexpr]
```
Definition at line 103 of file Galileo_INAV.h.

10.2.6.6.215 C_IC_LSB

```
double C_IC_LSB = TWO_N29  [constexpr]
```
Definition at line 129 of file GPS_L1_CA.h.

10.2.6.6.216 C_IS_4_LSB

```
double C_IS_4_LSB = TWO_N29  [constexpr]
```
Definition at line 105 of file Galileo_INAV.h.

10.2.6.6.217 C_IS_LSB

```
double C_IS_LSB = TWO_N29  [constexpr]
```
Definition at line 133 of file GPS_L1_CA.h.

10.2.6.6.218 C_RC_3_LSB

```
double C_RC_3_LSB = TWO_N5  [constexpr]
```
Definition at line 93 of file Galileo_INAV.h.

10.2.6.6.219 C_RC_LSB

```
double C_RC_LSB = TWO_N5  [constexpr]
```
Definition at line 137 of file GPS_L1_CA.h.

10.2.6.6.220 C_RS_3_LSB

```
double C_RS_3_LSB = TWO_N5  [constexpr]
```
Definition at line 95 of file Galileo_INAV.h.

10.2.6.6.221 C_RS_LSB

```
double C_RS_LSB = TWO_N5  [constexpr]
```
Definition at line 108 of file GPS_L1_CA.h.

10.2.6.6.222 C_UC_3_LSB

```
double C_UC_3_LSB = TWO_N29  [constexpr]
```
Definition at line 89 of file Galileo_INAV.h.

10.2.6.6.223 C_UC_LSB

```
double C_UC_LSB = TWO_N29  [constexpr]
```
Definition at line 114 of file GPS_L1_CA.h.

10.2.6.6.224 C_US_3_LSB

```
double C_US_3_LSB = TWO_N29  [constexpr]
```
Definition at line 91 of file Galileo_INAV.h.

10.2.6.6.225 C_US_LSB

```
double C_US_LSB = TWO_N29  [constexpr]
```
Definition at line 118 of file GPS_L1_CA.h.

10.2.6.6.226 CED_af0red_LSB

```
double CED_af0red_LSB = TWO_N26  [constexpr]
```
Definition at line 265 of file Galileo_INAV.h.

10.2.6.6.227 CED_af1red_LSB

```
double CED_af1red_LSB = TWO_N35  [constexpr]
```
Definition at line 267 of file Galileo_INAV.h.

10.2.6.6.228 CED_DeltaAred_LSB

```
double CED_DeltaAred_LSB = TWO_P8  [constexpr]
```
Definition at line 253 of file Galileo_INAV.h.

10.2.6.6.229 CED_Deltai0red_LSB

```
double CED_Deltai0red_LSB = TWO_N22  [constexpr]
```
Definition at line 259 of file Galileo_INAV.h.

10.2.6.6.230 CED_exred_LSB

```
double CED_exred_LSB = TWO_N22  [constexpr]
```
Definition at line 255 of file Galileo_INAV.h.

10.2.6.6.231 CED_eyred_LSB

```
double CED_eyred_LSB = TWO_N22  [constexpr]
```
Definition at line 257 of file Galileo_INAV.h.

10.2.6.6.232 CED_lambda0red_LSB

```
double CED_lambda0red_LSB = TWO_N22  [constexpr]
```
Definition at line 263 of file Galileo_INAV.h.

10.2.6.6.233 CED_Omega0red_LSB

```
double CED_Omega0red_LSB = TWO_N22  [constexpr]
```
Definition at line 261 of file Galileo_INAV.h.

10.2.6.6.234 CNAV_A0_LSB

```
double CNAV_A0_LSB = TWO_N35  [constexpr]
```
Definition at line 161 of file GPS_CNAV.h.

10.2.6.6.235 CNAV_A1_LSB

```
double CNAV_A1_LSB = TWO_N51  [constexpr]
```
Definition at line 163 of file GPS_CNAV.h.

10.2.6.6.236 CNAV_A2_LSB

```
double CNAV_A2_LSB = TWO_N68  [constexpr]
```
Definition at line 165 of file GPS_CNAV.h.

10.2.6.6.237 CNAV_A_DOT_LSB

```
double CNAV_A_DOT_LSB = TWO_N21  [constexpr]
```
Definition at line 58 of file GPS_CNAV.h.

10.2.6.6.238 CNAV_A_REF

```
double CNAV_A_REF = 26559710.0  [constexpr]
```
Definition at line 179 of file GPS_CNAV.h.

10.2.6.6.239 CNAV_AF0_LSB

```
double CNAV_AF0_LSB = TWO_N35  [constexpr]
```
Definition at line 107 of file GPS_CNAV.h.

10.2.6.6.240 CNAV_AF1_LSB

```
double CNAV_AF1_LSB = TWO_N48  [constexpr]
```
Definition at line 109 of file GPS_CNAV.h.

10.2.6.6.241 CNAV_AF2_LSB

```
double CNAV_AF2_LSB = TWO_N60  [constexpr]
```
Definition at line 111 of file GPS_CNAV.h.

10.2.6.6.242 CNAV_ALPHA0_LSB

```
double CNAV_ALPHA0_LSB = TWO_N30  [constexpr]
```
Definition at line 124 of file GPS_CNAV.h.

10.2.6.6.243 CNAV_ALPHA1_LSB

```
double CNAV_ALPHA1_LSB = TWO_N27  [constexpr]
```
Definition at line 126 of file GPS_CNAV.h.

10.2.6.6.244 CNAV_ALPHA2_LSB

```
double CNAV_ALPHA2_LSB = TWO_N24  [constexpr]
```
Definition at line 128 of file GPS_CNAV.h.

10.2.6.6.245 CNAV_ALPHA3_LSB

```
double CNAV_ALPHA3_LSB = TWO_N24  [constexpr]
```
Definition at line 130 of file GPS_CNAV.h.

10.2.6.6.246 CNAV_BETA0_LSB

```
double CNAV_BETA0_LSB = TWO_P11  [constexpr]
```
Definition at line 132 of file GPS_CNAV.h.

10.2.6.6.247 CNAV_BETA1_LSB

```
double CNAV_BETA1_LSB = TWO_P14  [constexpr]
```
Definition at line 134 of file GPS_CNAV.h.

10.2.6.6.248 CNAV_BETA2_LSB

```
double CNAV_BETA2_LSB = TWO_P16  [constexpr]
```
Definition at line 136 of file GPS_CNAV.h.

10.2.6.6.249 CNAV_BETA3_LSB

```
double CNAV_BETA3_LSB = TWO_P16  [constexpr]
```
Definition at line 138 of file GPS_CNAV.h.

10.2.6.6.250 CNAV_CIC_LSB

```
double CNAV_CIC_LSB = TWO_N30  [constexpr]
```
Definition at line 87 of file GPS_CNAV.h.

10.2.6.6.251 CNAV_CIS_LSB

```
double CNAV_CIS_LSB = TWO_N30  [constexpr]
```
Definition at line 85 of file GPS_CNAV.h.

10.2.6.6.252 CNAV_CRC_LSB

```
double CNAV_CRC_LSB = TWO_N8  [constexpr]
```
Definition at line 91 of file GPS_CNAV.h.

10.2.6.6.253 CNAV_CRS_LSB

```
double CNAV_CRS_LSB = TWO_N8  [constexpr]
```
Definition at line 89 of file GPS_CNAV.h.

10.2.6.6.254 CNAV_CUC_LSB

```
double CNAV_CUC_LSB = TWO_N30  [constexpr]
```
Definition at line 95 of file GPS_CNAV.h.

10.2.6.6.255 CNAV_CUS_LSB

```
double CNAV_CUS_LSB = TWO_N30  [constexpr]
```
Definition at line 93 of file GPS_CNAV.h.

10.2.6.6.256 CNAV_DELTA_A_LSB

```
double CNAV_DELTA_A_LSB = TWO_N9  [constexpr]
```
Definition at line 55 of file GPS_CNAV.h.

10.2.6.6.257 CNAV_DELTA_N0_DOT_LSB

```
double CNAV_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI  [constexpr]
```
Definition at line 63 of file GPS_CNAV.h.

10.2.6.6.258 CNAV_DELTA_N0_LSB

```
double CNAV_DELTA_N0_LSB = TWO_N44 * GNSS_PI  [constexpr]
```
Definition at line 61 of file GPS_CNAV.h.

10.2.6.6.259 CNAV_DELTA_OMEGA_DOT_LSB

```
double CNAV_DELTA_OMEGA_DOT_LSB = TWO_N44 * GNSS_PI  [constexpr]
```
Definition at line 81 of file GPS_CNAV.h.

10.2.6.6.260 CNAV_DELTA_TLS_LSB

```
int32_t CNAV_DELTA_TLS_LSB = 1  [constexpr]
```
Definition at line 167 of file GPS_CNAV.h.

10.2.6.6.261 CNAV_DELTA_TLSF_LSB

```
int32_t CNAV_DELTA_TLSF_LSB = 1  [constexpr]
```
Definition at line 177 of file GPS_CNAV.h.

10.2.6.6.262 CNAV_DELTA_UT1_GPS_DOT_LSB

```
double CNAV_DELTA_UT1_GPS_DOT_LSB = TWO_N25  [constexpr]
```
Definition at line 156 of file GPS_CNAV.h.

10.2.6.6.263 CNAV_DELTA_UT1_GPS_LSB

```
double CNAV_DELTA_UT1_GPS_LSB = TWO_N23  [constexpr]
```
Definition at line 154 of file GPS_CNAV.h.

10.2.6.6.264 CNAV_DN_LSB

```
int32_t CNAV_DN_LSB = 1  [constexpr]
```
Definition at line 175 of file GPS_CNAV.h.

10.2.6.6.265 CNAV_E_ECCENTRICITY_LSB

```
double CNAV_E_ECCENTRICITY_LSB = TWO_N34  [constexpr]
```
Definition at line 67 of file GPS_CNAV.h.

10.2.6.6.266 CNAV_I0_DOT_LSB

```
double CNAV_I0_DOT_LSB = TWO_N44 * GNSS_PI  [constexpr]
```
Definition at line 83 of file GPS_CNAV.h.

10.2.6.6.267 CNAV_I0_LSB

```
double CNAV_I0_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 79 of file GPS_CNAV.h.

10.2.6.6.268 CNAV_ISCL1_LSB

```
double CNAV_ISCL1_LSB = TWO_N35  [constexpr]
```
Definition at line 115 of file GPS_CNAV.h.

10.2.6.6.269 CNAV_ISCL2_LSB

```
double CNAV_ISCL2_LSB = TWO_N35  [constexpr]
```
Definition at line 117 of file GPS_CNAV.h.

10.2.6.6.270 CNAV_ISCL5I_LSB

```
double CNAV_ISCL5I_LSB = TWO_N35  [constexpr]
```
Definition at line 119 of file GPS_CNAV.h.

10.2.6.6.271 CNAV_ISCL5Q_LSB

```
double CNAV_ISCL5Q_LSB = TWO_N35  [constexpr]
```
Definition at line 121 of file GPS_CNAV.h.

10.2.6.6.272 CNAV_M0_LSB

```
double CNAV_M0_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 65 of file GPS_CNAV.h.

10.2.6.6.273 CNAV_OMEGA0_LSB

```
double CNAV_OMEGA0_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 77 of file GPS_CNAV.h.

10.2.6.6.274 CNAV_OMEGA_DOT_REF

```
double CNAV_OMEGA_DOT_REF = -2.6e-9  [constexpr]
```
Definition at line 180 of file GPS_CNAV.h.

10.2.6.6.275 CNAV_OMEGA_LSB

```
double CNAV_OMEGA_LSB = TWO_N32 * GNSS_PI  [constexpr]
```
Definition at line 69 of file GPS_CNAV.h.

10.2.6.6.276 CNAV_PM_X_DOT_LSB

```
double CNAV_PM_X_DOT_LSB = TWO_N21  [constexpr]
```
Definition at line 148 of file GPS_CNAV.h.

10.2.6.6.277 CNAV_PM_X_LSB

```
double CNAV_PM_X_LSB = TWO_N20  [constexpr]
```
Definition at line 146 of file GPS_CNAV.h.

10.2.6.6.278 CNAV_PM_Y_DOT_LSB

```
double CNAV_PM_Y_DOT_LSB = TWO_N21  [constexpr]
```
Definition at line 152 of file GPS_CNAV.h.

10.2.6.6.279 CNAV_PM_Y_LSB

```
double CNAV_PM_Y_LSB = TWO_N20  [constexpr]
```
Definition at line 150 of file GPS_CNAV.h.

10.2.6.6.280 CNAV_QZSS_A_REF

```
double CNAV_QZSS_A_REF = 42164200.0  [constexpr]
```
Definition at line 181 of file GPS_CNAV.h.

10.2.6.6.281 CNAV_T_EOP_LSB

```
int32_t CNAV_T_EOP_LSB = TWO_P4  [constexpr]
```
Definition at line 144 of file GPS_CNAV.h.

10.2.6.6.282 CNAV_TGD_LSB

```
double CNAV_TGD_LSB = TWO_N35  [constexpr]
```
Definition at line 113 of file GPS_CNAV.h.

10.2.6.6.283 CNAV_TOC_LSB

```
int32_t CNAV_TOC_LSB = 300  [constexpr]
```
Definition at line 105 of file GPS_CNAV.h.

10.2.6.6.284 CNAV_TOE1_LSB

```
int32_t CNAV_TOE1_LSB = 300  [constexpr]
```
Definition at line 52 of file GPS_CNAV.h.

10.2.6.6.285 CNAV_TOE2_LSB

```
int32_t CNAV_TOE2_LSB = 300  [constexpr]
```
Definition at line 75 of file GPS_CNAV.h.

10.2.6.6.286 CNAV_TOP1_LSB

```
int32_t CNAV_TOP1_LSB = 300  [constexpr]
```
Definition at line 48 of file GPS_CNAV.h.

10.2.6.6.287 CNAV_TOP2_LSB

```
int32_t CNAV_TOP2_LSB = 300  [constexpr]
```
Definition at line 100 of file GPS_CNAV.h.

10.2.6.6.288 CNAV_TOT_LSB

```
int32_t CNAV_TOT_LSB = TWO_P4  [constexpr]
```
Definition at line 169 of file GPS_CNAV.h.

10.2.6.6.289 CNAV_TOW_LSB

```
int32_t CNAV_TOW_LSB = 6  [constexpr]
```
Definition at line 41 of file GPS_CNAV.h.

10.2.6.6.290 CNAV_WN_LSF_LSB

```
int32_t CNAV_WN_LSF_LSB = 1  [constexpr]
```
Definition at line 173 of file GPS_CNAV.h.

10.2.6.6.291 CNAV_WN_OT_LSB

```
int32_t CNAV_WN_OT_LSB = 1  [constexpr]
```
Definition at line 171 of file GPS_CNAV.h.

10.2.6.6.292 CODE_L1A

```
uint32_t CODE_L1A = 10  [constexpr]
```
obs code: E1A (GAL)
Definition at line 40 of file gnss_obs_codes.h.

10.2.6.6.293 CODE_L1B

```
uint32_t CODE_L1B = 11  [constexpr]
```
obs code: E1B (GAL)
Definition at line 41 of file gnss_obs_codes.h.

10.2.6.6.294 CODE_L1C

```
uint32_t CODE_L1C = 1  [constexpr]
```
obs code: L1C/A,G1C/A,E1C (GPS,GLO,GAL,QZS,SBS)
Definition at line 31 of file gnss_obs_codes.h.

10.2.6.6.295 CODE_L1D

```
uint32_t CODE_L1D = 56  [constexpr]
```
obs code: B1C data (BDS)
Definition at line 86 of file gnss_obs_codes.h.

10.2.6.6.296 CODE_L1E

```
uint32_t CODE_L1E = 9  [constexpr]
```
(not used)
Definition at line 39 of file gnss_obs_codes.h.

10.2.6.6.297 CODE_L1I

```
uint32_t CODE_L1I = 47  [constexpr]
```
obs code: B1I (BDS)
Definition at line 77 of file gnss_obs_codes.h.

10.2.6.6.298 CODE_L1L

```
uint32_t CODE_L1L = 8  [constexpr]
```
obs code: L1C(P) (GPS,QZS)
Definition at line 38 of file gnss_obs_codes.h.

10.2.6.6.299 CODE_L1M

```
uint32_t CODE_L1M = 5  [constexpr]
```
obs code: L1M (GPS)
Definition at line 35 of file gnss_obs_codes.h.

10.2.6.6.300 CODE_L1N

```
uint32_t CODE_L1N = 6  [constexpr]
```
obs code: L1codeless (GPS)
Definition at line 36 of file gnss_obs_codes.h.

10.2.6.6.301 CODE_L1P

```
uint32_t CODE_L1P = 2  [constexpr]
```
obs code: L1P,G1P (GPS,GLO)
Definition at line 32 of file gnss_obs_codes.h.

10.2.6.6.302 CODE_L1Q

```
uint32_t CODE_L1Q = 48  [constexpr]
```
obs code: B1Q (BDS)
Definition at line 78 of file gnss_obs_codes.h.

10.2.6.6.303 CODE_L1S

```
uint32_t CODE_L1S = 7  [constexpr]
```
obs code: L1C(D) (GPS,QZS)
Definition at line 37 of file gnss_obs_codes.h.

10.2.6.6.304 CODE_L1W

```
uint32_t CODE_L1W = 3  [constexpr]
```
obs code: L1 Z-track (GPS)
Definition at line 33 of file gnss_obs_codes.h.

10.2.6.6.305 CODE_L1X

```
uint32_t CODE_L1X = 12  [constexpr]
```
obs code: E1B+C,L1C(D+P) (GAL,QZS)
Definition at line 42 of file gnss_obs_codes.h.

10.2.6.6.306 CODE_L1Y

```
uint32_t CODE_L1Y = 4  [constexpr]
```
obs code: L1Y (GPS)
Definition at line 34 of file gnss_obs_codes.h.

10.2.6.6.307 CODE_L1Z

```
uint32_t CODE_L1Z = 13  [constexpr]
```
obs code: E1A+B+C,L1SAIF (GAL,QZS)
Definition at line 43 of file gnss_obs_codes.h.

10.2.6.6.308 CODE_L2C

```
uint32_t CODE_L2C = 14  [constexpr]
```
obs code: L2C/A,G1C/A (GPS,GLO)
Definition at line 44 of file gnss_obs_codes.h.

10.2.6.6.309 CODE_L2D

```
uint32_t CODE_L2D = 15  [constexpr]
```
obs code: L2 L1C/A-(P2-P1) (GPS)
Definition at line 45 of file gnss_obs_codes.h.

10.2.6.6.310 CODE_L2I

```
uint32_t CODE_L2I = 40  [constexpr]
```
obs code: B1I (BDS)
Definition at line 70 of file gnss_obs_codes.h.

10.2.6.6.311 CODE_L2L

```
uint32_t CODE_L2L = 17  [constexpr]
```
obs code: L2C(L) (GPS,QZS)
Definition at line 47 of file gnss_obs_codes.h.

10.2.6.6.312 CODE_L2M

```
uint32_t CODE_L2M = 22  [constexpr]
```
obs code: L2M (GPS)
Definition at line 52 of file gnss_obs_codes.h.

10.2.6.6.313 CODE_L2N

```
uint32_t CODE_L2N = 23  [constexpr]
```
obs code: L2codeless (GPS)
Definition at line 53 of file gnss_obs_codes.h.

10.2.6.6.314 CODE_L2P

```
uint32_t CODE_L2P = 19  [constexpr]
```
obs code: L2P,G2P (GPS,GLO)
Definition at line 49 of file gnss_obs_codes.h.

10.2.6.6.315 CODE_L2Q

```
uint32_t CODE_L2Q = 41  [constexpr]
```
obs code: B1Q (BDS)
Definition at line 71 of file gnss_obs_codes.h.

10.2.6.6.316 CODE_L2S

```
uint32_t CODE_L2S = 16  [constexpr]
```
obs code: L2C(M) (GPS,QZS)
Definition at line 46 of file gnss_obs_codes.h.

10.2.6.6.317 CODE_L2W

```
uint32_t CODE_L2W = 20  [constexpr]
```
obs code: L2 Z-track (GPS)
Definition at line 50 of file gnss_obs_codes.h.

10.2.6.6.318 CODE_L2X

```
uint32_t CODE_L2X = 18  [constexpr]
```
obs code: L2C(M+L),B1I+Q (GPS,QZS,BDS)
Definition at line 48 of file gnss_obs_codes.h.

10.2.6.6.319 CODE_L2Y

```
uint32_t CODE_L2Y = 21  [constexpr]
```
obs code: L2Y (GPS)
Definition at line 51 of file gnss_obs_codes.h.

10.2.6.6.320 CODE_L3I

```
uint32_t CODE_L3I = 44  [constexpr]
```
obs code: G3I (GLO)
Definition at line 74 of file gnss_obs_codes.h.

10.2.6.6.321 CODE_L3Q

```
uint32_t CODE_L3Q = 45  [constexpr]
```
obs code: G3Q (GLO)
Definition at line 75 of file gnss_obs_codes.h.

10.2.6.6.322 CODE_L3X

```
uint32_t CODE_L3X = 46  [constexpr]
```
obs code: G3I+Q (GLO)
Definition at line 76 of file gnss_obs_codes.h.

10.2.6.6.323 CODE_L5A

```
uint32_t CODE_L5A = 49  [constexpr]
```
obs code: L5A SPS (IRN)
Definition at line 79 of file gnss_obs_codes.h.

10.2.6.6.324 CODE_L5B

```
uint32_t CODE_L5B = 50  [constexpr]
```
obs code: L5B RS(D) (IRN)
Definition at line 80 of file gnss_obs_codes.h.

10.2.6.6.325 CODE_L5C

```
uint32_t CODE_L5C = 51  [constexpr]
```
obs code: L5C RS(P) (IRN)
Definition at line 81 of file gnss_obs_codes.h.

10.2.6.6.326 CODE_L5D

```
uint32_t CODE_L5D = 57  [constexpr]
```
obs code: B2a data (BDS)
Definition at line 87 of file gnss_obs_codes.h.

10.2.6.6.327 CODE_L5I

```
uint32_t CODE_L5I = 24  [constexpr]
```
obs code: L5/E5aI (GPS,GAL,QZS,SBS)
Definition at line 54 of file gnss_obs_codes.h.

10.2.6.6.328 CODE_L5Q

```
uint32_t CODE_L5Q = 25  [constexpr]
```
obs code: L5/E5aQ (GPS,GAL,QZS,SBS)
Definition at line 55 of file gnss_obs_codes.h.

10.2.6.6.329 CODE_L5X

```
uint32_t CODE_L5X = 26  [constexpr]
```
obs code: L5/E5aI+Q/L5B+C (GPS,GAL,QZS,IRN,SBS)
Definition at line 56 of file gnss_obs_codes.h.

10.2.6.6.330 CODE_L6A

```
uint32_t CODE_L6A = 30  [constexpr]
```
obs code: E6A (GAL)
Definition at line 60 of file gnss_obs_codes.h.

10.2.6.6.331 CODE_L6B

```
uint32_t CODE_L6B = 31  [constexpr]
```
obs code: E6B (GAL)
Definition at line 61 of file gnss_obs_codes.h.

10.2.6.6.332 CODE_L6C

```
uint32_t CODE_L6C = 32  [constexpr]
```
obs code: E6C (GAL)
Definition at line 62 of file gnss_obs_codes.h.

10.2.6.6.333 CODE_L6I

```
uint32_t CODE_L6I = 42  [constexpr]
```
obs code: B3I (BDS)
Definition at line 72 of file gnss_obs_codes.h.

10.2.6.6.334 CODE_L6L

```
uint32_t CODE_L6L = 36  [constexpr]
```
obs code: LEXL (QZS)
Definition at line 66 of file gnss_obs_codes.h.

10.2.6.6.335 CODE_L6Q

```
uint32_t CODE_L6Q = 43  [constexpr]
```
obs code: B3Q (BDS)
Definition at line 73 of file gnss_obs_codes.h.

10.2.6.6.336 CODE_L6S

```
uint32_t CODE_L6S = 35  [constexpr]
```
obs code: LEXS (QZS)
Definition at line 65 of file gnss_obs_codes.h.

10.2.6.6.337 CODE_L6X

```
uint32_t CODE_L6X = 33  [constexpr]
```
obs code: E6B+C,LEXS+L,B3I+Q (GAL,QZS,BDS)
Definition at line 63 of file gnss_obs_codes.h.

10.2.6.6.338 CODE_L6Z

```
uint32_t CODE_L6Z = 34  [constexpr]
```
obs code: E6A+B+C (GAL)
Definition at line 64 of file gnss_obs_codes.h.

10.2.6.6.339 CODE_L7I

```
uint32_t CODE_L7I = 27  [constexpr]
```
obs code: E5bI,B2I (GAL,BDS)
Definition at line 57 of file gnss_obs_codes.h.

10.2.6.6.340 CODE_L7Q

```
uint32_t CODE_L7Q = 28  [constexpr]
```
obs code: E5bQ,B2Q (GAL,BDS)
Definition at line 58 of file gnss_obs_codes.h.

10.2.6.6.341 CODE_L7X

```
uint32_t CODE_L7X = 29  [constexpr]
```
obs code: E5bI+Q,B2I+Q (GAL,BDS)
Definition at line 59 of file gnss_obs_codes.h.

10.2.6.6.342 CODE_L8I

```
uint32_t CODE_L8I = 37  [constexpr]
```
obs code: E5(a+b)I (GAL)
Definition at line 67 of file gnss_obs_codes.h.

10.2.6.6.343 CODE_L8Q

```
uint32_t CODE_L8Q = 38  [constexpr]
```
obs code: E5(a+b)Q (GAL)
Definition at line 68 of file gnss_obs_codes.h.

10.2.6.6.344 CODE_L8X

```
uint32_t CODE_L8X = 39  [constexpr]
```
obs code: E5(a+b)I+Q (GAL)
Definition at line 69 of file gnss_obs_codes.h.

10.2.6.6.345 CODE_L9A

```
uint32_t CODE_L9A = 52  [constexpr]
```
obs code: SA SPS (IRN)
Definition at line 82 of file gnss_obs_codes.h.

10.2.6.6.346 CODE_L9B

```
uint32_t CODE_L9B = 53  [constexpr]
```
obs code: SB RS(D) (IRN)
Definition at line 83 of file gnss_obs_codes.h.

10.2.6.6.347 CODE_L9C

```
uint32_t CODE_L9C = 54  [constexpr]
```
obs code: SC RS(P) (IRN)
Definition at line 84 of file gnss_obs_codes.h.

10.2.6.6.348 CODE_L9X

```
uint32_t CODE_L9X = 55  [constexpr]
```
obs code: SB+C (IRN)
Definition at line 85 of file gnss_obs_codes.h.

10.2.6.6.349 CODE_NONE

```
uint32_t CODE_NONE = 0  [constexpr]
```
obs code: none or unknown
Definition at line 30 of file gnss_obs_codes.h.

10.2.6.6.350 D1_A0_ALMANAC_LSB

```
double D1_A0_ALMANAC_LSB = TWO_N20  [constexpr]
```
Definition at line 68 of file Beidou_DNAV.h.

10.2.6.6.351 D1_A0_LSB

```
double D1_A0_LSB = TWO_N33  [constexpr]
```
Definition at line 47 of file Beidou_DNAV.h.

10.2.6.6.352 D1_A0GAL_LSB

```
double D1_A0GAL_LSB = 0.1e-9  [constexpr]
```
Definition at line 78 of file Beidou_DNAV.h.

10.2.6.6.353 D1_A0GLO_LSB

```
double D1_A0GLO_LSB = 0.1e-9  [constexpr]
```
Definition at line 80 of file Beidou_DNAV.h.

10.2.6.6.354 D1_A0GPS_LSB

```
double D1_A0GPS_LSB = 0.1e-9  [constexpr]
```
Definition at line 76 of file Beidou_DNAV.h.

10.2.6.6.355 D1_A0UTC_LSB

```
double D1_A0UTC_LSB = TWO_N30  [constexpr]
```
Definition at line 82 of file Beidou_DNAV.h.

10.2.6.6.356 D1_A1_ALMANAC_LSB

```
double D1_A1_ALMANAC_LSB = TWO_N38  [constexpr]
```
Definition at line 67 of file Beidou_DNAV.h.

10.2.6.6.357 D1_A1_LSB

```
double D1_A1_LSB = TWO_N50  [constexpr]
```
Definition at line 48 of file Beidou_DNAV.h.

10.2.6.6.358 D1_A1GAL_LSB

```
double D1_A1GAL_LSB = 0.1e-9  [constexpr]
```
Definition at line 79 of file Beidou_DNAV.h.

10.2.6.6.359 D1_A1GLO_LSB

```
double D1_A1GLO_LSB = 0.1e-9  [constexpr]
```
Definition at line 81 of file Beidou_DNAV.h.

10.2.6.6.360 D1_A1GPS_LSB

```
double D1_A1GPS_LSB = 0.1e-9  [constexpr]
```
Definition at line 77 of file Beidou_DNAV.h.

10.2.6.6.361 D1_A1UTC_LSB

```
double D1_A1UTC_LSB = TWO_N50  [constexpr]
```
Definition at line 83 of file Beidou_DNAV.h.

10.2.6.6.362 D1_A2_LSB

```
double D1_A2_LSB = TWO_N66  [constexpr]
```
Definition at line 46 of file Beidou_DNAV.h.

10.2.6.6.363 D1_ALPHA0_LSB

```
double D1_ALPHA0_LSB = TWO_N30  [constexpr]
```
Definition at line 38 of file Beidou_DNAV.h.

10.2.6.6.364 D1_ALPHA1_LSB

```
double D1_ALPHA1_LSB = TWO_N27  [constexpr]
```
Definition at line 39 of file Beidou_DNAV.h.

10.2.6.6.365 D1_ALPHA2_LSB

```
double D1_ALPHA2_LSB = TWO_N24  [constexpr]
```
Definition at line 40 of file Beidou_DNAV.h.

10.2.6.6.366 D1_ALPHA3_LSB

```
double D1_ALPHA3_LSB = TWO_N24  [constexpr]
```
Definition at line 41 of file Beidou_DNAV.h.

10.2.6.6.367 D1_BETA0_LSB

```
double D1_BETA0_LSB = TWO_P11  [constexpr]
```
Definition at line 42 of file Beidou_DNAV.h.

10.2.6.6.368 D1_BETA1_LSB

```
double D1_BETA1_LSB = TWO_P14  [constexpr]
```
Definition at line 43 of file Beidou_DNAV.h.

10.2.6.6.369 D1_BETA2_LSB

```
double D1_BETA2_LSB = TWO_P16  [constexpr]
```
Definition at line 44 of file Beidou_DNAV.h.

10.2.6.6.370 D1_BETA3_LSB

```
double D1_BETA3_LSB = TWO_P16  [constexpr]
```
Definition at line 45 of file Beidou_DNAV.h.

10.2.6.6.371 D1_CIC_LSB

```
double D1_CIC_LSB = TWO_N31  [constexpr]
```
Definition at line 59 of file Beidou_DNAV.h.

10.2.6.6.372 D1_CIS_LSB

```
double D1_CIS_LSB = TWO_N31  [constexpr]
```
Definition at line 61 of file Beidou_DNAV.h.

10.2.6.6.373 D1_CRC_LSB

```
double D1_CRC_LSB = TWO_N6  [constexpr]
```
Definition at line 54 of file Beidou_DNAV.h.

10.2.6.6.374 D1_CRS_LSB

```
double D1_CRS_LSB = TWO_N6  [constexpr]
```
Definition at line 55 of file Beidou_DNAV.h.

10.2.6.6.375 D1_CUC_LSB

```
double D1_CUC_LSB = TWO_N31  [constexpr]
```
Definition at line 50 of file Beidou_DNAV.h.

10.2.6.6.376 D1_CUS_LSB

```
double D1_CUS_LSB = TWO_N31  [constexpr]
```
Definition at line 53 of file Beidou_DNAV.h.

10.2.6.6.377 D1_DELTA_I_LSB

```
double D1_DELTA_I_LSB = PI_TWO_N19  [constexpr]
```
Definition at line 71 of file Beidou_DNAV.h.

10.2.6.6.378 D1_DELTA_N_LSB

```
double D1_DELTA_N_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 49 of file Beidou_DNAV.h.

10.2.6.6.379 D1_E_ALMANAC_LSB

```
double D1_E_ALMANAC_LSB = TWO_N21  [constexpr]
```
Definition at line 70 of file Beidou_DNAV.h.

10.2.6.6.380 D1_E_LSB

```
double D1_E_LSB = TWO_N33  [constexpr]
```
Definition at line 52 of file Beidou_DNAV.h.

10.2.6.6.381 D1_I0_LSB

```
double D1_I0_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 58 of file Beidou_DNAV.h.

10.2.6.6.382 D1_IDOT_LSB

```
double D1_IDOT_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 62 of file Beidou_DNAV.h.

10.2.6.6.383 D1_M0_ALMANAC_LSB

```
double D1_M0_ALMANAC_LSB = PI_TWO_N23  [constexpr]
```
Definition at line 75 of file Beidou_DNAV.h.

10.2.6.6.384 D1_M0_LSB

```
double D1_M0_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 51 of file Beidou_DNAV.h.

10.2.6.6.385 D1_OMEGA0_ALMANAC_LSB

```
double D1_OMEGA0_ALMANAC_LSB = PI_TWO_N23  [constexpr]
```
Definition at line 69 of file Beidou_DNAV.h.

10.2.6.6.386 D1_OMEGA0_LSB

```
double D1_OMEGA0_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 63 of file Beidou_DNAV.h.

10.2.6.6.387 D1_OMEGA_ALMANAC_LSB

```
double D1_OMEGA_ALMANAC_LSB = PI_TWO_N23  [constexpr]
```
Definition at line 74 of file Beidou_DNAV.h.

10.2.6.6.388 D1_OMEGA_DOT_ALMANAC_LSB

```
double D1_OMEGA_DOT_ALMANAC_LSB = PI_TWO_N38  [constexpr]
```
Definition at line 73 of file Beidou_DNAV.h.

10.2.6.6.389 D1_OMEGA_DOT_LSB

```
double D1_OMEGA_DOT_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 60 of file Beidou_DNAV.h.

10.2.6.6.390 D1_OMEGA_LSB

```
double D1_OMEGA_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 64 of file Beidou_DNAV.h.

10.2.6.6.391 D1_SQRT_A_ALMANAC_LSB

```
double D1_SQRT_A_ALMANAC_LSB = TWO_N11  [constexpr]
```
Definition at line 66 of file Beidou_DNAV.h.

10.2.6.6.392 D1_SQRT_A_LSB

```
double D1_SQRT_A_LSB = TWO_N19  [constexpr]
```
Definition at line 56 of file Beidou_DNAV.h.

10.2.6.6.393 D1_TGD1_LSB

```
double D1_TGD1_LSB = 0.1e-9  [constexpr]
```
Definition at line 36 of file Beidou_DNAV.h.

10.2.6.6.394 D1_TGD2_LSB

```
double D1_TGD2_LSB = 0.1e-9  [constexpr]
```
Definition at line 37 of file Beidou_DNAV.h.

10.2.6.6.395 D1_TOA_LSB

```
double D1_TOA_LSB = TWO_P12  [constexpr]
```
Definition at line 72 of file Beidou_DNAV.h.

10.2.6.6.396 D1_TOC_LSB

```
double D1_TOC_LSB = TWO_P3  [constexpr]
```
Definition at line 35 of file Beidou_DNAV.h.

10.2.6.6.397 D1_TOE_LSB

```
double D1_TOE_LSB = TWO_P3  [constexpr]
```
Definition at line 57 of file Beidou_DNAV.h.

10.2.6.6.398 D2_DELTA_T_LSB

```
double D2_DELTA_T_LSB = 0.1  [constexpr]
```
Definition at line 84 of file Beidou_DNAV.h.

10.2.6.6.399 D2_ION_DELAY_LSB

```
double D2_ION_DELAY_LSB = 0.125  [constexpr]
```
Definition at line 85 of file Beidou_DNAV.h.

10.2.6.6.400 D2_ION_LOGICAL_FIRST_BIT

```
int32_t D2_ION_LOGICAL_FIRST_BIT = 51  [constexpr]
```
Definition at line 327 of file Beidou_DNAV.h.

10.2.6.6.401 D2_UDREI19_LOGICAL_FIRST_BIT

```
int32_t D2_UDREI19_LOGICAL_FIRST_BIT = 220  [constexpr]
```
Definition at line 326 of file Beidou_DNAV.h.

10.2.6.6.402 D2_UDREI1_LOGICAL_FIRST_BIT

```
int32_t D2_UDREI1_LOGICAL_FIRST_BIT = 172  [constexpr]
```
Definition at line 325 of file Beidou_DNAV.h.

10.2.6.6.403 D2R

```
double D2R = GNSS_PI / 180.0  [constexpr]
```
deg to rad
Definition at line 123 of file MATH_CONSTANTS.h.

10.2.6.6.404 DELTA_A_7_LSB

```
double DELTA_A_7_LSB = TWO_N9  [constexpr]
```
Definition at line 168 of file Galileo_INAV.h.

10.2.6.6.405 DELTA_A_8_LSB

```
double DELTA_A_8_LSB = TWO_N9  [constexpr]
```
Definition at line 193 of file Galileo_INAV.h.

10.2.6.6.406 DELTA_A_9_LSB

```
double DELTA_A_9_LSB = TWO_N9  [constexpr]
```
Definition at line 221 of file Galileo_INAV.h.

10.2.6.6.407 DELTA_I_7_LSB

```
double DELTA_I_7_LSB = TWO_N14  [constexpr]
```
Definition at line 174 of file Galileo_INAV.h.

10.2.6.6.408 DELTA_I_8_LSB

```
double DELTA_I_8_LSB = TWO_N14  [constexpr]
```
Definition at line 199 of file Galileo_INAV.h.

10.2.6.6.409 DELTA_I_9_LSB

```
double DELTA_I_9_LSB = TWO_N14  [constexpr]
```
Definition at line 227 of file Galileo_INAV.h.

10.2.6.6.410 DELTA_N_3_LSB

```
double DELTA_N_3_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 87 of file Galileo_INAV.h.

10.2.6.6.411 DELTA_N_LSB

```
double DELTA_N_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 110 of file GPS_L1_CA.h.

10.2.6.6.412 DELTAT_LS_LSB

```
double DELTAT_LS_LSB = 1  [constexpr]
```
Definition at line 178 of file GPS_L1_CA.h.

10.2.6.6.413 DELTAT_LSF_LSB

```
double DELTAT_LSF_LSB = 1  [constexpr]
```
Definition at line 184 of file GPS_L1_CA.h.

10.2.6.6.414 DFRQ1_GLO

```
double DFRQ1_GLO = 0.56250e6  [constexpr]
```
GLONASS G1 bias frequency (Hz/n).
Definition at line 39 of file gnss_frequencies.h.

10.2.6.6.415 DFRQ2_GLO

```
double DFRQ2_GLO = 0.43750e6  [constexpr]
```
GLONASS G2 bias frequency (Hz/n).
Definition at line 41 of file gnss_frequencies.h.

10.2.6.6.416 DN_LSB

```
double DN_LSB = 1  [constexpr]
```
Definition at line 182 of file GPS_L1_CA.h.

10.2.6.6.417 E_1_LSB

```
double E_1_LSB = TWO_N33  [constexpr]
```
Definition at line 64 of file Galileo_INAV.h.

10.2.6.6.418 E_7_LSB

```
double E_7_LSB = TWO_N16  [constexpr]
```
Definition at line 170 of file Galileo_INAV.h.

10.2.6.6.419 E_8_LSB

```
double E_8_LSB = TWO_N16  [constexpr]
```
Definition at line 195 of file Galileo_INAV.h.

10.2.6.6.420 E_9_LSB

```
double E_9_LSB = TWO_N16  [constexpr]
```
Definition at line 223 of file Galileo_INAV.h.

10.2.6.6.421 ECCENTRICITY_LSB

```
double ECCENTRICITY_LSB = TWO_N33  [constexpr]
```
Definition at line 116 of file GPS_L1_CA.h.

10.2.6.6.422 FIRST_RS_BIT

```
int32_t FIRST_RS_BIT = 7  [constexpr]
```
Definition at line 277 of file Galileo_INAV.h.

10.2.6.6.423 FIRST_RS_BIT_AFTER_IODNAV

```
int32_t FIRST_RS_BIT_AFTER_IODNAV = 17  [constexpr]
```
Definition at line 278 of file Galileo_INAV.h.

10.2.6.6.424 FNAV_A0_4_LSB

```
double FNAV_A0_4_LSB = TWO_N30  [constexpr]
```
Definition at line 109 of file Galileo_FNAV.h.

10.2.6.6.425 FNAV_A0G_4_LSB

```
double FNAV_A0G_4_LSB = TWO_N35  [constexpr]
```
Definition at line 122 of file Galileo_FNAV.h.

10.2.6.6.426 FNAV_A12_2_LSB

```
double FNAV_A12_2_LSB = TWO_N19  [constexpr]
```
Definition at line 73 of file Galileo_FNAV.h.

10.2.6.6.427 FNAV_A1_4_LSB

```
double FNAV_A1_4_LSB = TWO_N50  [constexpr]
```
Definition at line 111 of file Galileo_FNAV.h.

10.2.6.6.428 FNAV_A1G_4_LSB

```
double FNAV_A1G_4_LSB = TWO_N51  [constexpr]
```
Definition at line 124 of file Galileo_FNAV.h.

10.2.6.6.429 FNAV_AF0_1_LSB

```
double FNAV_AF0_1_LSB = TWO_N34  [constexpr]
```
Definition at line 40 of file Galileo_FNAV.h.

10.2.6.6.430 FNAV_AF0_5_LSB

```
double FNAV_AF0_5_LSB = TWO_N19  [constexpr]
```
Definition at line 149 of file Galileo_FNAV.h.

10.2.6.6.431 FNAV_AF1_1_LSB

```
double FNAV_AF1_1_LSB = TWO_N46  [constexpr]
```
Definition at line 42 of file Galileo_FNAV.h.

10.2.6.6.432 FNAV_AF1_5_LSB

```
double FNAV_AF1_5_LSB = TWO_N38  [constexpr]
```
Definition at line 151 of file Galileo_FNAV.h.

10.2.6.6.433 FNAV_AF2_1_LSB

```
double FNAV_AF2_1_LSB = TWO_N59  [constexpr]
```
Definition at line 44 of file Galileo_FNAV.h.

10.2.6.6.434 FNAV_AI0_1_LSB

```
double FNAV_AI0_1_LSB = TWO_N2  [constexpr]
```
Definition at line 47 of file Galileo_FNAV.h.

10.2.6.6.435 FNAV_AI1_1_LSB

```
double FNAV_AI1_1_LSB = TWO_N8  [constexpr]
```
Definition at line 49 of file Galileo_FNAV.h.

10.2.6.6.436 FNAV_AI2_1_LSB

```
double FNAV_AI2_1_LSB = TWO_N15  [constexpr]
```
Definition at line 51 of file Galileo_FNAV.h.

10.2.6.6.437 FNAV_BGD_1_LSB

```
double FNAV_BGD_1_LSB = TWO_N32  [constexpr]
```
Definition at line 58 of file Galileo_FNAV.h.

10.2.6.6.438 FNAV_CIC_4_LSB

```
double FNAV_CIC_4_LSB = TWO_N29  [constexpr]
```
Definition at line 105 of file Galileo_FNAV.h.

10.2.6.6.439 FNAV_CIS_4_LSB

```
double FNAV_CIS_4_LSB = TWO_N29  [constexpr]
```
Definition at line 107 of file Galileo_FNAV.h.

10.2.6.6.440 FNAV_CRC_3_LSB

```
double FNAV_CRC_3_LSB = TWO_N5  [constexpr]
```
Definition at line 94 of file Galileo_FNAV.h.

10.2.6.6.441 FNAV_CRS_3_LSB

```
double FNAV_CRS_3_LSB = TWO_N5  [constexpr]
```
Definition at line 96 of file Galileo_FNAV.h.

10.2.6.6.442 FNAV_CUC_3_LSB

```
double FNAV_CUC_3_LSB = TWO_N29  [constexpr]
```
Definition at line 90 of file Galileo_FNAV.h.

10.2.6.6.443 FNAV_CUS_3_LSB

```
double FNAV_CUS_3_LSB = TWO_N29  [constexpr]
```
Definition at line 92 of file Galileo_FNAV.h.

10.2.6.6.444 FNAV_DELTAA12_5_LSB

```
double FNAV_DELTAA12_5_LSB = TWO_N9  [constexpr]
```
Definition at line 135 of file Galileo_FNAV.h.

10.2.6.6.445 FNAV_DELTAI_5_LSB

```
double FNAV_DELTAI_5_LSB = TWO_N14  [constexpr]
```
Definition at line 141 of file Galileo_FNAV.h.

10.2.6.6.446 FNAV_DELTAN_3_LSB

```
double FNAV_DELTAN_3_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 88 of file Galileo_FNAV.h.

10.2.6.6.447 FNAV_E_2_LSB

```
double FNAV_E_2_LSB = TWO_N33  [constexpr]
```
Definition at line 71 of file Galileo_FNAV.h.

10.2.6.6.448 FNAV_E_5_LSB

```
double FNAV_E_5_LSB = TWO_N16  [constexpr]
```
Definition at line 137 of file Galileo_FNAV.h.

10.2.6.6.449 FNAV_I0_3_LSB

```
double FNAV_I0_3_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 84 of file Galileo_FNAV.h.

10.2.6.6.450 FNAV_IDOT_2_LSB

```
double FNAV_IDOT_2_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 77 of file Galileo_FNAV.h.

10.2.6.6.451 FNAV_M0_2_LSB

```
double FNAV_M0_2_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 67 of file Galileo_FNAV.h.

10.2.6.6.452 FNAV_M0_5_LSB

```
double FNAV_M0_5_LSB = TWO_N15  [constexpr]
```
Definition at line 147 of file Galileo_FNAV.h.

10.2.6.6.453 FNAV_OMEGA0_2_LSB

```
double FNAV_OMEGA0_2_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 75 of file Galileo_FNAV.h.

10.2.6.6.454 FNAV_OMEGA0_5_LSB

```
double FNAV_OMEGA0_5_LSB = TWO_N15  [constexpr]
```
Definition at line 143 of file Galileo_FNAV.h.

10.2.6.6.455 FNAV_OMEGADOT_2_LSB

```
double FNAV_OMEGADOT_2_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 69 of file Galileo_FNAV.h.

10.2.6.6.456 FNAV_OMEGADOT_5_LSB

```
double FNAV_OMEGADOT_5_LSB = TWO_N33  [constexpr]
```
Definition at line 145 of file Galileo_FNAV.h.

10.2.6.6.457 FNAV_T0A_5_LSB

```
int32_t FNAV_T0A_5_LSB = 600  [constexpr]
```
Definition at line 132 of file Galileo_FNAV.h.

10.2.6.6.458 FNAV_T0C_1_LSB

```
int32_t FNAV_T0C_1_LSB = 60  [constexpr]
```
Definition at line 38 of file Galileo_FNAV.h.

10.2.6.6.459 FNAV_T0E_3_LSB

```
int32_t FNAV_T0E_3_LSB = 60  [constexpr]
```
Definition at line 98 of file Galileo_FNAV.h.

10.2.6.6.460 FNAV_T0G_4_LSB

```
int32_t FNAV_T0G_4_LSB = 3600  [constexpr]
```
Definition at line 120 of file Galileo_FNAV.h.

10.2.6.6.461 FNAV_T0T_4_LSB

```
int32_t FNAV_T0T_4_LSB = 3600  [constexpr]
```
Definition at line 114 of file Galileo_FNAV.h.

10.2.6.6.462 FNAV_W_3_LSB

```
double FNAV_W_3_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 86 of file Galileo_FNAV.h.

10.2.6.6.463 FNAV_W_5_LSB

```
double FNAV_W_5_LSB = TWO_N15  [constexpr]
```
Definition at line 139 of file Galileo_FNAV.h.

10.2.6.6.464 FREQ1

```
double FREQ1 = 1.57542e9  [constexpr]
```
L1/E1 frequency (Hz).
Definition at line 31 of file gnss_frequencies.h.

10.2.6.6.465 FREQ1_BDS

```
double FREQ1_BDS = 1.561098e9  [constexpr]
```
BeiDou B1 frequency (Hz).
Definition at line 43 of file gnss_frequencies.h.

10.2.6.6.466 FREQ1_GLO

```
double FREQ1_GLO = 1.60200e9  [constexpr]
```
GLONASS G1 base frequency (Hz).
Definition at line 38 of file gnss_frequencies.h.

10.2.6.6.467 FREQ2

```
double FREQ2 = 1.22760e9  [constexpr]
```
L2 frequency (Hz).
Definition at line 32 of file gnss_frequencies.h.

10.2.6.6.468 FREQ2_BDS

```
double FREQ2_BDS = 1.20714e9  [constexpr]
```
BeiDou B2 frequency (Hz).
Definition at line 44 of file gnss_frequencies.h.

10.2.6.6.469 FREQ2_GLO

```
double FREQ2_GLO = 1.24600e9  [constexpr]
```
GLONASS G2 base frequency (Hz).
Definition at line 40 of file gnss_frequencies.h.

10.2.6.6.470 FREQ3_BDS

```
double FREQ3_BDS = 1.26852e9  [constexpr]
```
BeiDou B3 frequency (Hz).
Definition at line 45 of file gnss_frequencies.h.

10.2.6.6.471 FREQ3_GLO

```
double FREQ3_GLO = 1.202025e9  [constexpr]
```
GLONASS G3 frequency (Hz).
Definition at line 42 of file gnss_frequencies.h.

10.2.6.6.472 FREQ5

```
double FREQ5 = 1.17645e9  [constexpr]
```
L5/E5a frequency (Hz).
Definition at line 33 of file gnss_frequencies.h.

10.2.6.6.473 FREQ6

```
double FREQ6 = 1.27875e9  [constexpr]
```
E6/LEX frequency (Hz).
Definition at line 34 of file gnss_frequencies.h.

10.2.6.6.474 FREQ7

```
double FREQ7 = 1.20714e9  [constexpr]
```
E5b frequency (Hz).
Definition at line 35 of file gnss_frequencies.h.

10.2.6.6.475 FREQ8

```
double FREQ8 = 1.191795e9  [constexpr]
```
E5a+b frequency (Hz).
Definition at line 36 of file gnss_frequencies.h.

10.2.6.6.476 FREQ9

```
double FREQ9 = 2.492028e9  [constexpr]
```
S frequency (Hz).
Definition at line 37 of file gnss_frequencies.h.

10.2.6.6.477 GALILEO_CNAV_BITS_FOR_CRC

```
int32_t GALILEO_CNAV_BITS_FOR_CRC = GALILEO_CNAV_HAS_PAGE_DATA_BITS + GALILEO_CNAV_PAGE_↩
RESERVED_BITS  [constexpr]
```
Definition at line 85 of file Galileo_CNAV.h.

10.2.6.6.478 GALILEO_CNAV_BYTES_FOR_CRC

```
int32_t GALILEO_CNAV_BYTES_FOR_CRC = 58  [constexpr]
```
Definition at line 75 of file Galileo_CNAV.h.

10.2.6.6.479 GALILEO_CNAV_CRC_LENGTH

```
int32_t GALILEO_CNAV_CRC_LENGTH = 24  [constexpr]
```
Definition at line 76 of file Galileo_CNAV.h.

10.2.6.6.480 GALILEO_CNAV_HAS_PAGE_DATA_BITS

```
int32_t GALILEO_CNAV_HAS_PAGE_DATA_BITS = 448  [constexpr]
```
Definition at line 73 of file Galileo_CNAV.h.

10.2.6.6.481 GALILEO_CNAV_INFORMATION_VECTOR_LENGTH

```
int32_t GALILEO_CNAV_INFORMATION_VECTOR_LENGTH = 32  [constexpr]
```
Definition at line 83 of file Galileo_CNAV.h.

10.2.6.6.482 GALILEO_CNAV_INTERLEAVER_COLS

```
int32_t GALILEO_CNAV_INTERLEAVER_COLS = 123  [constexpr]
```
Definition at line 71 of file Galileo_CNAV.h.

10.2.6.6.483 GALILEO_CNAV_INTERLEAVER_ROWS

```
int32_t GALILEO_CNAV_INTERLEAVER_ROWS = 8  [constexpr]
```
Definition at line 70 of file Galileo_CNAV.h.

10.2.6.6.484 GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK

```
int32_t GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK = 255  [constexpr]
```
Definition at line 80 of file Galileo_CNAV.h.

10.2.6.6.485 GALILEO_CNAV_MESSAGE_BITS_PER_PAGE

```
int32_t GALILEO_CNAV_MESSAGE_BITS_PER_PAGE = 424  [constexpr]
```
Definition at line 77 of file Galileo_CNAV.h.

10.2.6.6.486 GALILEO_CNAV_MT1_HEADER_BITS

```
int32_t GALILEO_CNAV_MT1_HEADER_BITS = 32  [constexpr]
```
Definition at line 81 of file Galileo_CNAV.h.

10.2.6.6.487 GALILEO_CNAV_OCTETS_IN_SUBPAGE

```
int32_t GALILEO_CNAV_OCTETS_IN_SUBPAGE = 53  [constexpr]
```
Definition at line 82 of file Galileo_CNAV.h.

10.2.6.6.488 GALILEO_CNAV_PAGE_HEADER_BITS

```
int32_t GALILEO_CNAV_PAGE_HEADER_BITS = 24  [constexpr]
```
Definition at line 78 of file Galileo_CNAV.h.

10.2.6.6.489 GALILEO_CNAV_PAGE_MS

```
int32_t GALILEO_CNAV_PAGE_MS = 1  [constexpr]
```
Definition at line 69 of file Galileo_CNAV.h.

10.2.6.6.490 GALILEO_CNAV_PAGE_RESERVED_BITS

```
int32_t GALILEO_CNAV_PAGE_RESERVED_BITS = 14  [constexpr]
```
Definition at line 74 of file Galileo_CNAV.h.

10.2.6.6.491 GALILEO_CNAV_PREAMBLE

```
char GALILEO_CNAV_PREAMBLE[17] = "1011011101110000"  [constexpr]
```
Definition at line 106 of file Galileo_CNAV.h.

10.2.6.6.492 GALILEO_CNAV_PREAMBLE_LENGTH_BITS

```
int32_t GALILEO_CNAV_PREAMBLE_LENGTH_BITS = 16  [constexpr]
```
Definition at line 79 of file Galileo_CNAV.h.

10.2.6.6.493 GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS

```
int32_t GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS = 1000  [constexpr]
```
Definition at line 68 of file Galileo_CNAV.h.

10.2.6.6.494 GALILEO_CNAV_SYMBOLS_PER_PAGE

```
int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000  [constexpr]
```
Definition at line 67 of file Galileo_CNAV.h.

10.2.6.6.495 GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND

```
int32_t GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND = 1000  [constexpr]
```
Definition at line 72 of file Galileo_CNAV.h.

10.2.6.6.496 GALILEO_DATA_FRAME_BITS

```
int32_t GALILEO_DATA_FRAME_BITS = 196  [constexpr]
```
Definition at line 47 of file Galileo_INAV.h.

10.2.6.6.497 GALILEO_DATA_FRAME_BYTES

```
int32_t GALILEO_DATA_FRAME_BYTES = 25  [constexpr]
```
Definition at line 48 of file Galileo_INAV.h.

10.2.6.6.498 GALILEO_DATA_JK_BITS

```
int32_t GALILEO_DATA_JK_BITS = 128  [constexpr]
```
Definition at line 46 of file Galileo_INAV.h.

10.2.6.6.499 GALILEO_E1_B_CODE_LENGTH_CHIPS

```
double GALILEO_E1_B_CODE_LENGTH_CHIPS = 4092.0  [constexpr]
```
Galileo E1-B code length [chips].
Definition at line 39 of file Galileo_E1.h.

10.2.6.6.500 GALILEO_E1_B_PRIMARY_CODE

```
char GALILEO_E1_B_PRIMARY_CODE[GALILEO_E1_NUMBER_OF_CODES][1024]  [constexpr]
```
Definition at line 56 of file Galileo_E1.h.

10.2.6.6.501 GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH = 1023  [constexpr]
```
Definition at line 55 of file Galileo_E1.h.

10.2.6.6.502 GALILEO_E1_B_SAMPLES_PER_SYMBOL

```
int32_t GALILEO_E1_B_SAMPLES_PER_SYMBOL = 1  [constexpr]
```
(Galileo_E1_CODE_CHIP_RATE_HZ / Galileo_E1_B_CODE_LENGTH_CHIPS) / Galileo_E1_B_SYMBOL_↩ RATE_BPS
Definition at line 42 of file Galileo_E1.h.

10.2.6.6.503 GALILEO_E1_B_SYMBOL_RATE_BPS

```
double GALILEO_E1_B_SYMBOL_RATE_BPS = 250.0  [constexpr]
```
Galileo E1-B symbol rate [bits/second].
Definition at line 40 of file Galileo_E1.h.

10.2.6.6.504 GALILEO_E1_C_PRIMARY_CODE

```
char GALILEO_E1_C_PRIMARY_CODE[GALILEO_E1_NUMBER_OF_CODES][1024]  [constexpr]
```
Definition at line 760 of file Galileo_E1.h.

10.2.6.6.505 GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH = 1023  [constexpr]
```
Definition at line 759 of file Galileo_E1.h.

10.2.6.6.506 GALILEO_E1_C_SECONDARY_CODE

```
char GALILEO_E1_C_SECONDARY_CODE[26] = "00111000000010101101100 10"  [constexpr]
```
Definition at line 52 of file Galileo_E1.h.

10.2.6.6.507 GALILEO_E1_C_SECONDARY_CODE_LENGTH

```
int32_t GALILEO_E1_C_SECONDARY_CODE_LENGTH = 25  [constexpr]
```
Galileo E1-C secondary code length [chips].
Definition at line 43 of file Galileo_E1.h.

10.2.6.6.508 GALILEO_E1_CODE_CHIP_RATE_CPS

```
double GALILEO_E1_CODE_CHIP_RATE_CPS = 1.023e6  [constexpr]
```
Galileo E1 code rate [chips/s].
Definition at line 35 of file Galileo_E1.h.

10.2.6.6.509 GALILEO_E1_CODE_PERIOD_MS

```
uint32_t GALILEO_E1_CODE_PERIOD_MS = 4  [constexpr]
```
Galileo E1 code period [ms].
Definition at line 41 of file Galileo_E1.h.

10.2.6.6.510 GALILEO_E1_CODE_PERIOD_S

```
double GALILEO_E1_CODE_PERIOD_S = 0.004  [constexpr]
```
Galileo E1 code period [s].
Definition at line 36 of file Galileo_E1.h.

10.2.6.6.511 GALILEO_E1_FREQ_HZ

```
double GALILEO_E1_FREQ_HZ = FREQ1  [constexpr]
```
Galileo E1 carrier frequency [Hz].
Definition at line 34 of file Galileo_E1.h.

10.2.6.6.512 GALILEO_E1_HISTORY_DEEP

```
int32_t GALILEO_E1_HISTORY_DEEP = 100  [constexpr]
```
Observable history length for interpotalion.
Definition at line 50 of file Galileo_E1.h.

10.2.6.6.513 GALILEO_E1_NUMBER_OF_CODES

```
int32_t GALILEO_E1_NUMBER_OF_CODES = 50  [constexpr]
```
Definition at line 44 of file Galileo_E1.h.

10.2.6.6.514 GALILEO_E1_OPT_ACQ_FS_SPS

```
uint32_t GALILEO_E1_OPT_ACQ_FS_SPS = 2000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 48 of file Galileo_E1.h.

10.2.6.6.515 GALILEO_E1_SUB_CARRIER_A_RATE_HZ

```
double GALILEO_E1_SUB_CARRIER_A_RATE_HZ = 1.023e6  [constexpr]
```
Galileo E1 sub-carrier 'a' rate [Hz].
Definition at line 37 of file Galileo_E1.h.

10.2.6.6.516 GALILEO_E1_SUB_CARRIER_B_RATE_HZ

```
double GALILEO_E1_SUB_CARRIER_B_RATE_HZ = 6.138e6  [constexpr]
```
Galileo E1 sub-carrier 'b' rate [Hz].
Definition at line 38 of file Galileo_E1.h.

10.2.6.6.517 GALILEO_E5A_CODE_CHIP_RATE_CPS

```
double GALILEO_E5A_CODE_CHIP_RATE_CPS = 1.023e7  [constexpr]
```
Galileo E5a code rate [chips/s].
Definition at line 33 of file Galileo_E5a.h.

10.2.6.6.518 GALILEO_E5A_CODE_LENGTH_CHIPS

```
int32_t GALILEO_E5A_CODE_LENGTH_CHIPS = 10230  [constexpr]
```

Galileo E5a primary code length [chips].
Definition at line 37 of file Galileo_E5a.h.

10.2.6.6.519 GALILEO_E5A_CODE_PERIOD_MS

```
int32_t GALILEO_E5A_CODE_PERIOD_MS = 1  [constexpr]
```

Galileo E5a primary code period [ms].
Definition at line 40 of file Galileo_E5a.h.

10.2.6.6.520 GALILEO_E5A_CODE_PERIOD_S

```
double GALILEO_E5A_CODE_PERIOD_S = 0.001  [constexpr]
```

Galileo E5a primary code period [s].
Definition at line 36 of file Galileo_E5a.h.

10.2.6.6.521 GALILEO_E5A_CRC_ERROR_LIMIT

```
int32_t GALILEO_E5A_CRC_ERROR_LIMIT = 6  [constexpr]
```

Definition at line 47 of file Galileo_E5a.h.

10.2.6.6.522 GALILEO_E5A_FREQ_HZ

```
double GALILEO_E5A_FREQ_HZ = FREQ5  [constexpr]
```

Galileo E5a carrier frequency [Hz].
Definition at line 32 of file Galileo_E5a.h.

10.2.6.6.523 GALILEO_E5A_HISTORY_DEEP

```
int32_t GALILEO_E5A_HISTORY_DEEP = 20  [constexpr]
```

Definition at line 46 of file Galileo_E5a.h.

10.2.6.6.524 GALILEO_E5A_I_PRIMARY_CODE

```
char GALILEO_E5A_I_PRIMARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][2559]  [constexpr]
```

Definition at line 72 of file Galileo_E5a.h.

10.2.6.6.525 GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH = 2558  [constexpr]
```

Definition at line 71 of file Galileo_E5a.h.

10.2.6.6.526 GALILEO_E5A_I_SECONDARY_CODE

```
char GALILEO_E5A_I_SECONDARY_CODE[] = "10000100001011101001"  [constexpr]
```

Definition at line 3581 of file Galileo_E5a.h.

10.2.6.6.527 GALILEO_E5A_I_SECONDARY_CODE_LENGTH

```
int32_t GALILEO_E5A_I_SECONDARY_CODE_LENGTH = 20  [constexpr]
```

Galileo E5a-I secondary code length [chips].
Definition at line 38 of file Galileo_E5a.h.

10.2.6.6.528 GALILEO_E5A_I_TIERED_CODE_PERIOD_S

```
double GALILEO_E5A_I_TIERED_CODE_PERIOD_S = 0.020  [constexpr]
```

Galileo E5a-I tiered code period [s].
Definition at line 34 of file Galileo_E5a.h.

10.2.6.6.529 GALILEO_E5A_NUMBER_OF_CODES

```
int32_t GALILEO_E5A_NUMBER_OF_CODES = 50  [constexpr]
```
Definition at line 42 of file Galileo_E5a.h.

10.2.6.6.530 GALILEO_E5A_OPT_ACQ_FS_SPS

```
uint32_t GALILEO_E5A_OPT_ACQ_FS_SPS = 10000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 50 of file Galileo_E5a.h.

10.2.6.6.531 GALILEO_E5A_Q_PRIMARY_CODE

```
char GALILEO_E5A_Q_PRIMARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][2559]  [constexpr]
```
Definition at line 1827 of file Galileo_E5a.h.

10.2.6.6.532 GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH = 2558  [constexpr]
```
Definition at line 1826 of file Galileo_E5a.h.

10.2.6.6.533 GALILEO_E5A_Q_SECONDARY_CODE

```
char GALILEO_E5A_Q_SECONDARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][101]  [constexpr]
```
Definition at line 3585 of file Galileo_E5a.h.

10.2.6.6.534 GALILEO_E5A_Q_SECONDARY_CODE_LENGTH

```
int32_t GALILEO_E5A_Q_SECONDARY_CODE_LENGTH = 100  [constexpr]
```
Galileo E5a-Q secondary code length [chips].
Definition at line 39 of file Galileo_E5a.h.

10.2.6.6.535 GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH

```
size_t GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH = 100  [constexpr]
```
Definition at line 3584 of file Galileo_E5a.h.

10.2.6.6.536 GALILEO_E5A_Q_TIERED_CODE_PERIOD_S

```
double GALILEO_E5A_Q_TIERED_CODE_PERIOD_S = 0.100  [constexpr]
```
Galileo E5a-Q tiered code period [s].
Definition at line 35 of file Galileo_E5a.h.

10.2.6.6.537 GALILEO_E5A_SYMBOL_RATE_BPS

```
int32_t GALILEO_E5A_SYMBOL_RATE_BPS = 50  [constexpr]
```
Galileo E5a symbol rate [bits/second].
Definition at line 41 of file Galileo_E5a.h.

10.2.6.6.538 GALILEO_E5B_CODE_CHIP_RATE_CPS

```
double GALILEO_E5B_CODE_CHIP_RATE_CPS = 1.023e7  [constexpr]
```
Galileo E5b code rate [chips/s].
Definition at line 34 of file Galileo_E5b.h.

10.2.6.6.539 GALILEO_E5B_CODE_LENGTH_CHIPS

```
int32_t GALILEO_E5B_CODE_LENGTH_CHIPS = 10230  [constexpr]
```
Galileo E5b primary code length [chips].
Definition at line 39 of file Galileo_E5b.h.

10.2.6.6.540 GALILEO_E5B_CODE_PERIOD_MS

```
int32_t GALILEO_E5B_CODE_PERIOD_MS = 1  [constexpr]
```
Galileo E5b primary code period [ms].
Definition at line 38 of file Galileo_E5b.h.

10.2.6.6.541 GALILEO_E5B_CODE_PERIOD_S

```
double GALILEO_E5B_CODE_PERIOD_S = 0.001  [constexpr]
```
Galileo E5b primary code period [s].
Definition at line 37 of file Galileo_E5b.h.

10.2.6.6.542 GALILEO_E5B_FREQ_HZ

```
double GALILEO_E5B_FREQ_HZ = FREQ7  [constexpr]
```
Galileo E5b carrier frequency [Hz].
Definition at line 33 of file Galileo_E5b.h.

10.2.6.6.543 GALILEO_E5B_HISTORY_DEEP

```
int32_t GALILEO_E5B_HISTORY_DEEP = 100  [constexpr]
```
Definition at line 47 of file Galileo_E5b.h.

10.2.6.6.544 GALILEO_E5B_I_PRIMARY_CODE

```
char GALILEO_E5B_I_PRIMARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][2559]  [constexpr]
```
Definition at line 57 of file Galileo_E5b.h.

10.2.6.6.545 GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH = 2558  [constexpr]
```
Definition at line 56 of file Galileo_E5b.h.

10.2.6.6.546 GALILEO_E5B_I_SECONDARY_CODE

```
char GALILEO_E5B_I_SECONDARY_CODE[5] = "1110"  [constexpr]
```
Definition at line 53 of file Galileo_E5b.h.

10.2.6.6.547 GALILEO_E5B_I_SECONDARY_CODE_LENGTH

```
int32_t GALILEO_E5B_I_SECONDARY_CODE_LENGTH = 4  [constexpr]
```
Galileo E5b-I secondary code length [chips].
Definition at line 40 of file Galileo_E5b.h.

10.2.6.6.548 GALILEO_E5B_I_TIERED_CODE_PERIOD_S

```
double GALILEO_E5B_I_TIERED_CODE_PERIOD_S = 0.004  [constexpr]
```
Galileo E5b-I tiered code period [s].
Definition at line 35 of file Galileo_E5b.h.

10.2.6.6.549 GALILEO_E5B_NUMBER_OF_CODES

```
int32_t GALILEO_E5B_NUMBER_OF_CODES = 50  [constexpr]
```
Definition at line 43 of file Galileo_E5b.h.

10.2.6.6.550 GALILEO_E5B_OPT_ACQ_FS_SPS

```
uint32_t GALILEO_E5B_OPT_ACQ_FS_SPS = 10000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 50 of file Galileo_E5b.h.

10.2.6.6.551 GALILEO_E5B_Q_PRIMARY_CODE

```
char GALILEO_E5B_Q_PRIMARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][2559]  [constexpr]
```
Definition at line 1812 of file Galileo_E5b.h.

10.2.6.6.552 GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH = 2558  [constexpr]
```
Definition at line 1811 of file Galileo_E5b.h.

10.2.6.6.553 GALILEO_E5B_Q_SECONDARY_CODE

```
char GALILEO_E5B_Q_SECONDARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][101]  [constexpr]
```
Definition at line 3568 of file Galileo_E5b.h.

10.2.6.6.554 GALILEO_E5B_Q_SECONDARY_CODE_LENGTH

```
int32_t GALILEO_E5B_Q_SECONDARY_CODE_LENGTH = 100  [constexpr]
```
Galileo E5b-Q secondary code length [chips].
Definition at line 41 of file Galileo_E5b.h.

10.2.6.6.555 GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH

```
size_t GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH = 100  [constexpr]
```
Definition at line 3567 of file Galileo_E5b.h.

10.2.6.6.556 GALILEO_E5B_Q_TIERED_CODE_PERIOD_S

```
double GALILEO_E5B_Q_TIERED_CODE_PERIOD_S = 0.100  [constexpr]
```
Galileo E5b-Q tiered code period [s].
Definition at line 36 of file Galileo_E5b.h.

10.2.6.6.557 GALILEO_E5B_SYMBOL_RATE_BPS

```
int32_t GALILEO_E5B_SYMBOL_RATE_BPS = 250  [constexpr]
```
Galileo E5b symbol rate [bits/second].
Definition at line 42 of file Galileo_E5b.h.

10.2.6.6.558 GALILEO_E6_B_CODE_CHIP_RATE_CPS

```
double GALILEO_E6_B_CODE_CHIP_RATE_CPS = 5.115e6  [constexpr]
```
Galileo E6 B code rate [chips/s].
Definition at line 31 of file Galileo_E6.h.

10.2.6.6.559 GALILEO_E6_B_CODE_LENGTH_CHIPS

```
double GALILEO_E6_B_CODE_LENGTH_CHIPS = 5115.0  [constexpr]
```
Galileo E6 B code length [chips].
Definition at line 35 of file Galileo_E6.h.

10.2.6.6.560 GALILEO_E6_B_PRIMARY_CODE

```
char GALILEO_E6_B_PRIMARY_CODE[GALILEO_E6_NUMBER_OF_CODES][1280]  [constexpr]
```
Definition at line 45 of file Galileo_E6.h.

10.2.6.6.561 GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH = 1279  [constexpr]
```
Definition at line 44 of file Galileo_E6.h.

10.2.6.6.562 GALILEO_E6_C_CODE_CHIP_RATE_CPS

```
double GALILEO_E6_C_CODE_CHIP_RATE_CPS = 5.115e6  [constexpr]
```
Galileo E6 C code rate [chips/s].
Definition at line 32 of file Galileo_E6.h.

10.2.6.6.563 GALILEO_E6_C_CODE_LENGTH_CHIPS

```
double GALILEO_E6_C_CODE_LENGTH_CHIPS = 5115.0  [constexpr]
```
Galileo E6 C code length [chips].
Definition at line 36 of file Galileo_E6.h.

10.2.6.6.564 GALILEO_E6_C_PRIMARY_CODE

```
char GALILEO_E6_C_PRIMARY_CODE[GALILEO_E6_NUMBER_OF_CODES][1280]  [constexpr]
```
Definition at line 950 of file Galileo_E6.h.

10.2.6.6.565 GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH

```
size_t GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH = 1279  [constexpr]
```
Definition at line 949 of file Galileo_E6.h.

10.2.6.6.566 GALILEO_E6_C_SECONDARY_CODE

```
char GALILEO_E6_C_SECONDARY_CODE[GALILEO_E6_NUMBER_OF_CODES][26]  [constexpr]
```
Definition at line 1855 of file Galileo_E6.h.

10.2.6.6.567 GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS

```
double GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS = 100.0  [constexpr]
```
Galileo E6 C secondary code length [chips].
Definition at line 37 of file Galileo_E6.h.

10.2.6.6.568 GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH

```
size_t GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH = 25  [constexpr]
```
Definition at line 1854 of file Galileo_E6.h.

10.2.6.6.569 GALILEO_E6_CODE_PERIOD_MS

```
uint32_t GALILEO_E6_CODE_PERIOD_MS = 1  [constexpr]
```
Galileo E& B/C code period [ms].
Definition at line 38 of file Galileo_E6.h.

10.2.6.6.570 GALILEO_E6_CODE_PERIOD_S

```
double GALILEO_E6_CODE_PERIOD_S = 0.001  [constexpr]
```
Galileo E6 code period [s].
Definition at line 33 of file Galileo_E6.h.

10.2.6.6.571 GALILEO_E6_FREQ_HZ

```
double GALILEO_E6_FREQ_HZ = FREQ6  [constexpr]
```
Galileo E6 carrier frequency [Hz].
Definition at line 30 of file Galileo_E6.h.

10.2.6.6.572 GALILEO_E6_NUMBER_OF_CODES

```
int32_t GALILEO_E6_NUMBER_OF_CODES = 50  [constexpr]
```
Definition at line 40 of file Galileo_E6.h.

10.2.6.6.573 GALILEO_E6_OPT_ACQ_FS_SPS

```
uint32_t GALILEO_E6_OPT_ACQ_FS_SPS = 10000000  [constexpr]
```
Definition at line 42 of file Galileo_E6.h.

10.2.6.6.574 GALILEO_F

```
double GALILEO_F = -4.442807309e-10  [constexpr]
```
Constant, [s/(m)^(1/2)]. OS SIS ICD v2.0, pag. 47.
Definition at line 36 of file MATH_CONSTANTS.h.

10.2.6.6.575 GALILEO_FNAV_CODES_PER_PAGE

```
int32_t GALILEO_FNAV_CODES_PER_PAGE = 10000  [constexpr]
```
Definition at line 59 of file Galileo_E5a.h.

10.2.6.6.576 GALILEO_FNAV_CODES_PER_PREAMBLE

```
int32_t GALILEO_FNAV_CODES_PER_PREAMBLE = 240  [constexpr]
```
Definition at line 56 of file Galileo_E5a.h.

10.2.6.6.577 GALILEO_FNAV_CODES_PER_SYMBOL

```
int32_t GALILEO_FNAV_CODES_PER_SYMBOL = 20  [constexpr]
```
Definition at line 55 of file Galileo_E5a.h.

10.2.6.6.578 GALILEO_FNAV_DATA_FRAME_BITS

```
int32_t GALILEO_FNAV_DATA_FRAME_BITS = 214  [constexpr]
```
Definition at line 65 of file Galileo_E5a.h.

10.2.6.6.579 GALILEO_FNAV_DATA_FRAME_BYTES

```
int32_t GALILEO_FNAV_DATA_FRAME_BYTES = 27  [constexpr]
```
Definition at line 66 of file Galileo_E5a.h.

10.2.6.6.580 GALILEO_FNAV_INTERLEAVER_COLS

```
int32_t GALILEO_FNAV_INTERLEAVER_COLS = 61  [constexpr]
```
Definition at line 62 of file Galileo_E5a.h.

10.2.6.6.581 GALILEO_FNAV_INTERLEAVER_ROWS

```
int32_t GALILEO_FNAV_INTERLEAVER_ROWS = 8  [constexpr]
```
Definition at line 61 of file Galileo_E5a.h.

10.2.6.6.582 GALILEO_FNAV_PAGE_TYPE_BITS

```
int32_t GALILEO_FNAV_PAGE_TYPE_BITS = 6  [constexpr]
```
Definition at line 63 of file Galileo_E5a.h.

10.2.6.6.583 GALILEO_FNAV_PREAMBLE

```
char GALILEO_FNAV_PREAMBLE[13] = "101101110000"  [constexpr]
```
Definition at line 68 of file Galileo_E5a.h.

10.2.6.6.584 GALILEO_FNAV_PREAMBLE_LENGTH_BITS

```
int32_t GALILEO_FNAV_PREAMBLE_LENGTH_BITS = 12  [constexpr]
```
Definition at line 54 of file Galileo_E5a.h.

10.2.6.6.585 GALILEO_FNAV_SECONDS_PER_PAGE

```
int32_t GALILEO_FNAV_SECONDS_PER_PAGE = 10  [constexpr]
```
Definition at line 58 of file Galileo_E5a.h.

10.2.6.6.586 GALILEO_FNAV_SYMBOLS_PER_PAGE

```
int32_t GALILEO_FNAV_SYMBOLS_PER_PAGE = 500  [constexpr]
```
Definition at line 57 of file Galileo_E5a.h.

10.2.6.6.587 GALILEO_GM

```
double GALILEO_GM = 3.986004418e14  [constexpr]
```
Geocentric gravitational constant[m^3/s^2], OS SIS ICD v2.0, pag. 44.
Definition at line 35 of file MATH_CONSTANTS.h.

10.2.6.6.588 GALILEO_HAS_HALF_WEEK_SECONDS

```
uint32_t GALILEO_HAS_HALF_WEEK_SECONDS = GALILEO_HAS_SECONDS_PER_WEEK / 2U  [constexpr]
```
Definition at line 62 of file Galileo_CNAV.h.

10.2.6.6.589 GALILEO_HAS_INVALID_TOW

```
uint32_t GALILEO_HAS_INVALID_TOW = std::numeric_limits<uint32_t>::max()  [constexpr]
```
Definition at line 64 of file Galileo_CNAV.h.

10.2.6.6.590 GALILEO_HAS_INVALID_WEEK

```
uint32_t GALILEO_HAS_INVALID_WEEK = std::numeric_limits<uint32_t>::max()  [constexpr]
```
Definition at line 63 of file Galileo_CNAV.h.

10.2.6.6.591 GALILEO_HAS_SECONDS_PER_HOUR

```
uint32_t GALILEO_HAS_SECONDS_PER_HOUR = 3600U  [constexpr]
```
Definition at line 60 of file Galileo_CNAV.h.

10.2.6.6.592 GALILEO_HAS_SECONDS_PER_WEEK

```
uint32_t GALILEO_HAS_SECONDS_PER_WEEK = 604800U  [constexpr]
```
Definition at line 61 of file Galileo_CNAV.h.

10.2.6.6.593 GALILEO_INAV_DUMMY_WORD_TYPE

```
int32_t GALILEO_INAV_DUMMY_WORD_TYPE = 63  [constexpr]
```
Definition at line 51 of file Galileo_INAV.h.

10.2.6.6.594 GALILEO_INAV_ENCODED_SSP1

```
char GALILEO_INAV_ENCODED_SSP1[17] = "1110100100100101"  [constexpr]
```
Definition at line 306 of file Galileo_INAV.h.

10.2.6.6.595 GALILEO_INAV_ENCODED_SSP2

```
char GALILEO_INAV_ENCODED_SSP2[17] = "0110110001001110"  [constexpr]
```
Definition at line 307 of file Galileo_INAV.h.

10.2.6.6.596 GALILEO_INAV_ENCODED_SSP3

```
char GALILEO_INAV_ENCODED_SSP3[17] = "1101000000111110"  [constexpr]
```
Definition at line 308 of file Galileo_INAV.h.

10.2.6.6.597 GALILEO_INAV_EVEN_PAGE_TYPE_BIT

```
size_t GALILEO_INAV_EVEN_PAGE_TYPE_BIT = 1  [constexpr]
```
Definition at line 49 of file Galileo_INAV.h.

10.2.6.6.598 GALILEO_INAV_INTERLEAVER_COLS

```
int32_t GALILEO_INAV_INTERLEAVER_COLS = 30  [constexpr]
```
Definition at line 43 of file Galileo_INAV.h.

10.2.6.6.599 GALILEO_INAV_INTERLEAVER_ROWS

```
int32_t GALILEO_INAV_INTERLEAVER_ROWS = 8  [constexpr]
```
Definition at line 42 of file Galileo_INAV.h.

10.2.6.6.600 GALILEO_INAV_ODD_PAGE_TYPE_BIT

```
size_t GALILEO_INAV_ODD_PAGE_TYPE_BIT = 115  [constexpr]
```
Definition at line 50 of file Galileo_INAV.h.

10.2.6.6.601 GALILEO_INAV_PAGE_PART_MS

```
int32_t GALILEO_INAV_PAGE_PART_MS = 1000  [constexpr]
```
Definition at line 40 of file Galileo_INAV.h.

10.2.6.6.602 GALILEO_INAV_PAGE_PART_SECONDS

```
int32_t GALILEO_INAV_PAGE_PART_SECONDS = 1  [constexpr]
```
Definition at line 39 of file Galileo_INAV.h.

10.2.6.6.603 GALILEO_INAV_PAGE_PART_SYMBOLS

```
int32_t GALILEO_INAV_PAGE_PART_SYMBOLS = 250  [constexpr]
```
Each Galileo INAV pages are composed of two parts (even and odd) each of 250 symbols, including preamble. See Galileo ICD 4.3.2.
Definition at line 38 of file Galileo_INAV.h.

10.2.6.6.604 GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS

```
double GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS = 2.04  [constexpr]
```
Page Duration + (Galileo I/NAV Preamble bits)∗(Galileo E5b-I tiered Code Period(seconds)).
Definition at line 34 of file Galileo_INAV.h.

10.2.6.6.605 GALILEO_INAV_PAGE_SECONDS

```
int32_t GALILEO_INAV_PAGE_SECONDS = 2  [constexpr]
```
Definition at line 41 of file Galileo_INAV.h.

10.2.6.6.606 GALILEO_INAV_PAGE_SYMBOLS

```
uint32_t GALILEO_INAV_PAGE_SYMBOLS = 500  [constexpr]
```
The complete Galileo INAV page length.
Definition at line 35 of file Galileo_INAV.h.

10.2.6.6.607 GALILEO_INAV_PLAIN_SSP1

```
char GALILEO_INAV_PLAIN_SSP1[9] = "00000100"  [constexpr]
```
Definition at line 303 of file Galileo_INAV.h.

10.2.6.6.608 GALILEO_INAV_PLAIN_SSP2

```
char GALILEO_INAV_PLAIN_SSP2[9] = "00101011"  [constexpr]
```
Definition at line 304 of file Galileo_INAV.h.

10.2.6.6.609 GALILEO_INAV_PLAIN_SSP3

```
char GALILEO_INAV_PLAIN_SSP3[9] = "00101111"  [constexpr]
```
Definition at line 305 of file Galileo_INAV.h.

10.2.6.6.610 GALILEO_INAV_PREAMBLE

```
char GALILEO_INAV_PREAMBLE[11] = "0101100000"  [constexpr]
```
Definition at line 52 of file Galileo_INAV.h.

10.2.6.6.611 GALILEO_INAV_PREAMBLE_LENGTH_BITS

```
int32_t GALILEO_INAV_PREAMBLE_LENGTH_BITS = 10  [constexpr]
```
Definition at line 36 of file Galileo_INAV.h.

10.2.6.6.612 GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS

```
int32_t GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS = 250  [constexpr]
```
Definition at line 37 of file Galileo_INAV.h.

10.2.6.6.613 GALILEO_ISM_CRC_DATA_BITS

```
int32_t GALILEO_ISM_CRC_DATA_BITS = 96  [constexpr]
```
Definition at line 294 of file Galileo_INAV.h.

10.2.6.6.614 GALILEO_ISM_CRC_DATA_BYTES

```
int32_t GALILEO_ISM_CRC_DATA_BYTES = 12  [constexpr]
```
Definition at line 295 of file Galileo_INAV.h.

10.2.6.6.615 GALILEO_PAGE_TYPE_BITS

```
int32_t GALILEO_PAGE_TYPE_BITS = 6  [constexpr]
```
Definition at line 45 of file Galileo_INAV.h.

10.2.6.6.616 GALILEO_TELEMETRY_RATE_BITS_SECOND

```
int32_t GALILEO_TELEMETRY_RATE_BITS_SECOND = 250  [constexpr]
```
Definition at line 44 of file Galileo_INAV.h.

10.2.6.6.617 GLONASS_C20

```
double GLONASS_C20 = -1082.63e-6  [constexpr]
```
Second zonal coefficient of spherical harmonic expansion.
Definition at line 45 of file GLONASS_L1_L2_CA.h.

10.2.6.6.618 GLONASS_CA_NBR_SATS

```
int32_t GLONASS_CA_NBR_SATS = 24  [constexpr]
```
Definition at line 88 of file GLONASS_L1_L2_CA.h.

10.2.6.6.619 GLONASS_EARTH_INCLINATION

```
double GLONASS_EARTH_INCLINATION = 0.000409148809899e3  [constexpr]
```
Mean inclination of ecliptic to equator (23 deg 26 min 33 sec) [rad].
Definition at line 47 of file GLONASS_L1_L2_CA.h.

10.2.6.6.620 GLONASS_EARTH_RADIUS

```
double GLONASS_EARTH_RADIUS = 6378.136  [constexpr]
```
Equatorial radius of Earth [km].
Definition at line 46 of file GLONASS_L1_L2_CA.h.

10.2.6.6.621 GLONASS_F_M_A

```
double GLONASS_F_M_A = 0.35e9  [constexpr]
```
Gravitational constant of atmosphere [m^3/s^2].
Definition at line 35 of file GLONASS_L1_L2_CA.h.

10.2.6.6.622 GLONASS_FLATTENING

```
double GLONASS_FLATTENING = 1.0 / 29825784.0  [constexpr]
```
Flattening parameter.
Definition at line 37 of file GLONASS_L1_L2_CA.h.

10.2.6.6.623 GLONASS_GM

```
double GLONASS_GM = 398600.44e9  [constexpr]
```
Universal gravitational constant times the mass of the Earth, [m^3/s^2].
Definition at line 40 of file MATH_CONSTANTS.h.

10.2.6.6.624 GLONASS_GNAV_CRC_I_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_I_INDEX {9, 10, 12, 13, 15, 17, 19, 20, 22, 24,
26, 28, 30, 32, 34, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 66, 68,
70, 72, 74, 76, 78, 80, 82, 84}
```
Definition at line 261 of file GLONASS_L1_L2_CA.h.

10.2.6.6.625 GLONASS_GNAV_CRC_J_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_J_INDEX {9, 11, 12, 14, 15, 18, 19, 21, 22, 25,
26, 29, 30, 33, 34, 36, 37, 40, 41, 44, 45, 48, 49, 52, 53, 56, 57, 60, 61, 64, 65, 67, 68,
71, 72, 75, 76, 79, 80, 83, 84}
```
Definition at line 262 of file GLONASS_L1_L2_CA.h.

10.2.6.6.626 GLONASS_GNAV_CRC_K_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_K_INDEX {10, 11, 12, 16, 17, 18, 19, 23, 24, 25,
26, 31, 32, 33, 34, 38, 39, 40, 41, 46, 47, 48, 49, 54, 55, 56, 57, 62, 63, 64, 65, 69, 70,
71, 72, 77, 78, 79, 80, 85}
```
Definition at line 263 of file GLONASS_L1_L2_CA.h.

10.2.6.6.627 GLONASS_GNAV_CRC_L_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_L_INDEX {13, 14, 15, 16, 17, 18, 19, 27, 28, 29,
30, 31, 32, 33, 34, 42, 43, 44, 45, 46, 47, 48, 49, 58, 59, 60, 61, 62, 63, 64, 65, 73, 74,
75, 76, 77, 78, 79, 80}
```
Definition at line 264 of file GLONASS_L1_L2_CA.h.

10.2.6.6.628 GLONASS_GNAV_CRC_M_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_M_INDEX {20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
30, 31, 32, 33, 34, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 81, 82,
83, 84, 85}
```
Definition at line 265 of file GLONASS_L1_L2_CA.h.

10.2.6.6.629 GLONASS_GNAV_CRC_N_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_N_INDEX {35, 36, 37, 38, 39, 40, 41, 42, 43, 44,
45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65}
```

Definition at line 266 of file GLONASS_L1_L2_CA.h.

10.2.6.6.630 GLONASS_GNAV_CRC_P_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_P_INDEX {66, 67, 68, 69, 70, 71, 72, 73, 74, 75,
76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
```

Definition at line 267 of file GLONASS_L1_L2_CA.h.

10.2.6.6.631 GLONASS_GNAV_CRC_Q_INDEX

```
const std::vector<int32_t> GLONASS_GNAV_CRC_Q_INDEX {9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41,
42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64,
65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
```

Definition at line 268 of file GLONASS_L1_L2_CA.h.

10.2.6.6.632 GLONASS_GNAV_DATA_SYMBOLS

```
int32_t GLONASS_GNAV_DATA_SYMBOLS = 1700  [constexpr]
```

Definition at line 117 of file GLONASS_L1_L2_CA.h.

10.2.6.6.633 GLONASS_GNAV_ECC_LOCATOR

```
const std::vector<int32_t> GLONASS_GNAV_ECC_LOCATOR {0, 0, 1, 8, 2, 9, 10, 11, 3, 12, 13, 14,
15, 16, 17, 18, 4, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 5, 34, 35, 36,
37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 6, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82,
83, 84}
```

Definition at line 269 of file GLONASS_L1_L2_CA.h.

10.2.6.6.634 GLONASS_GNAV_HAMMING_CODE_BITS

```
int32_t GLONASS_GNAV_HAMMING_CODE_BITS = 8  [constexpr]
```

Number of bits in hamming code sequence of GNAV message.
Definition at line 116 of file GLONASS_L1_L2_CA.h.

10.2.6.6.635 GLONASS_GNAV_PREAMBLE_DURATION_S

```
double GLONASS_GNAV_PREAMBLE_DURATION_S = 0.300  [constexpr]
```

Definition at line 103 of file GLONASS_L1_L2_CA.h.

10.2.6.6.636 GLONASS_GNAV_PREAMBLE_LENGTH_BITS

```
int32_t GLONASS_GNAV_PREAMBLE_LENGTH_BITS = 30  [constexpr]
```

Definition at line 105 of file GLONASS_L1_L2_CA.h.

10.2.6.6.637 GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS

```
int32_t GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS = 300  [constexpr]
```

Definition at line 106 of file GLONASS_L1_L2_CA.h.

10.2.6.6.638 GLONASS_GNAV_PREAMBLE_PERIOD_BITS

```
int32_t GLONASS_GNAV_PREAMBLE_PERIOD_BITS = 200  [constexpr]
```

Definition at line 108 of file GLONASS_L1_L2_CA.h.

10.2.6.6.639 GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS

```
int32_t GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS = 2000  [constexpr]
```
Definition at line 107 of file GLONASS_L1_L2_CA.h.

10.2.6.6.640 GLONASS_GNAV_PREAMBLE_STR

```
char GLONASS_GNAV_PREAMBLE_STR[301] = "1111111111111111111111111111111111111111111111111111000000000000000000000
[constexpr]
```
Definition at line 102 of file GLONASS_L1_L2_CA.h.

10.2.6.6.641 GLONASS_GNAV_SECONDS_PER_WEEK

```
double GLONASS_GNAV_SECONDS_PER_WEEK = 604800.0  [constexpr]
```
Definition at line 104 of file GLONASS_L1_L2_CA.h.

10.2.6.6.642 GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE

```
int32_t GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE = 200  [constexpr]
```
Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
Definition at line 114 of file GLONASS_L1_L2_CA.h.

10.2.6.6.643 GLONASS_GNAV_STRING_BITS

```
int32_t GLONASS_GNAV_STRING_BITS = 85  [constexpr]
```
Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
Definition at line 115 of file GLONASS_L1_L2_CA.h.

10.2.6.6.644 GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE

```
int32_t GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE = 2000  [constexpr]
```
Definition at line 113 of file GLONASS_L1_L2_CA.h.

10.2.6.6.645 GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND

```
int32_t GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND = 50  [constexpr]
```
NAV message bit rate [bits/s].
Definition at line 109 of file GLONASS_L1_L2_CA.h.

10.2.6.6.646 GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND

```
int32_t GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND = GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND *
GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT  [constexpr]
```
NAV message bit rate [symbols/s].
Definition at line 112 of file GLONASS_L1_L2_CA.h.

10.2.6.6.647 GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT = 10  [constexpr]
```
Definition at line 110 of file GLONASS_L1_L2_CA.h.

10.2.6.6.648 GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT

```
int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT = 10  [constexpr]
```
Definition at line 111 of file GLONASS_L1_L2_CA.h.

10.2.6.6.649 GLONASS_GRAVITY

```
double GLONASS_GRAVITY = 97803284.0  [constexpr]
```
Equatorial acceleration of gravity [mGal].
Definition at line 38 of file GLONASS_L1_L2_CA.h.

10.2.6.6.650 GLONASS_GRAVITY_CORRECTION

```
double GLONASS_GRAVITY_CORRECTION = 0.87  [constexpr]
```
Correction to acceleration of gravity at sea-level due to Atmosphere[uGal].
Definition at line 39 of file GLONASS_L1_L2_CA.h.

10.2.6.6.651 GLONASS_J2

```
double GLONASS_J2 = 1082625.75e-9  [constexpr]
```
Second zonal harmonic of the geopotential.
Definition at line 40 of file GLONASS_L1_L2_CA.h.

10.2.6.6.652 GLONASS_J4

```
double GLONASS_J4 = -2370.89e-9  [constexpr]
```
Fourth zonal harmonic of the geopotential.
Definition at line 41 of file GLONASS_L1_L2_CA.h.

10.2.6.6.653 GLONASS_J6

```
double GLONASS_J6 = 6.08e-9  [constexpr]
```
Sixth zonal harmonic of the geopotential.
Definition at line 42 of file GLONASS_L1_L2_CA.h.

10.2.6.6.654 GLONASS_J8

```
double GLONASS_J8 = 1.40e-11  [constexpr]
```
Eighth zonal harmonic of the geopotential.
Definition at line 43 of file GLONASS_L1_L2_CA.h.

10.2.6.6.655 GLONASS_L1_CA_CHIP_PERIOD_S

```
double GLONASS_L1_CA_CHIP_PERIOD_S = 1.9569e-06  [constexpr]
```
GLONASS L1 C/A chip period [seconds].
Definition at line 83 of file GLONASS_L1_L2_CA.h.

10.2.6.6.656 GLONASS_L1_CA_CODE_LENGTH_CHIPS

```
double GLONASS_L1_CA_CODE_LENGTH_CHIPS = 511.0  [constexpr]
```
GLONASS L1 C/A code length [chips].
Definition at line 80 of file GLONASS_L1_L2_CA.h.

10.2.6.6.657 GLONASS_L1_CA_CODE_PERIOD_MS

```
uint32_t GLONASS_L1_CA_CODE_PERIOD_MS = 1  [constexpr]
```
Definition at line 86 of file GLONASS_L1_L2_CA.h.

10.2.6.6.658 GLONASS_L1_CA_CODE_PERIOD_S

```
double GLONASS_L1_CA_CODE_PERIOD_S = 0.001  [constexpr]
```
GLONASS L1 C/A code period [seconds].
Definition at line 81 of file GLONASS_L1_L2_CA.h.

10.2.6.6.659 GLONASS_L1_CA_CODE_RATE_CPS

```
double GLONASS_L1_CA_CODE_RATE_CPS = 0.511e6  [constexpr]
```
GLONASS L1 C/A code rate [chips/s].
Definition at line 79 of file GLONASS_L1_L2_CA.h.

10.2.6.6.660 GLONASS_L1_CA_DFREQ_HZ

```
double GLONASS_L1_CA_DFREQ_HZ = DFRQ1_GLO  [constexpr]
```

Freq Bias for GLONASS L1 [Hz].
Definition at line 78 of file GLONASS_L1_L2_CA.h.

10.2.6.6.661 GLONASS_L1_CA_FREQ_HZ

```
double GLONASS_L1_CA_FREQ_HZ = FREQ1_GLO  [constexpr]
```

L1 [Hz].
Definition at line 77 of file GLONASS_L1_L2_CA.h.

10.2.6.6.662 GLONASS_L1_CA_HISTORY_DEEP

```
int32_t GLONASS_L1_CA_HISTORY_DEEP = 100  [constexpr]
```

Definition at line 91 of file GLONASS_L1_L2_CA.h.

10.2.6.6.663 GLONASS_L1_CA_SYMBOL_PERIOD_S

```
double GLONASS_L1_CA_SYMBOL_PERIOD_S = 0.01  [constexpr]
```

GLONASS L1 C/A symbol period [seconds].
Definition at line 82 of file GLONASS_L1_L2_CA.h.

10.2.6.6.664 GLONASS_L1_CA_SYMBOL_RATE_BPS

```
double GLONASS_L1_CA_SYMBOL_RATE_BPS = 1000.0  [constexpr]
```

Definition at line 84 of file GLONASS_L1_L2_CA.h.

10.2.6.6.665 GLONASS_L2_CA_CHIP_PERIOD_S

```
double GLONASS_L2_CA_CHIP_PERIOD_S = 1.9569e-06  [constexpr]
```

GLONASS L1 C/A chip period [seconds].
Definition at line 74 of file GLONASS_L1_L2_CA.h.

10.2.6.6.666 GLONASS_L2_CA_CODE_LENGTH_CHIPS

```
double GLONASS_L2_CA_CODE_LENGTH_CHIPS = 511.0  [constexpr]
```

GLONASS L1 C/A code length [chips].
Definition at line 71 of file GLONASS_L1_L2_CA.h.

10.2.6.6.667 GLONASS_L2_CA_CODE_PERIOD_MS

```
uint32_t GLONASS_L2_CA_CODE_PERIOD_MS = 1  [constexpr]
```

Definition at line 87 of file GLONASS_L1_L2_CA.h.

10.2.6.6.668 GLONASS_L2_CA_CODE_PERIOD_S

```
double GLONASS_L2_CA_CODE_PERIOD_S = 0.001  [constexpr]
```

GLONASS L1 C/A code period [seconds].
Definition at line 72 of file GLONASS_L1_L2_CA.h.

10.2.6.6.669 GLONASS_L2_CA_CODE_RATE_CPS

```
double GLONASS_L2_CA_CODE_RATE_CPS = 0.511e6  [constexpr]
```

GLONASS L1 C/A code rate [chips/s].
Definition at line 70 of file GLONASS_L1_L2_CA.h.

10.2.6.6.670 GLONASS_L2_CA_DFREQ_HZ

```
double GLONASS_L2_CA_DFREQ_HZ = DFRQ2_GLO  [constexpr]
```

Freq Bias for GLONASS L1 [Hz].
Definition at line 69 of file GLONASS_L1_L2_CA.h.

10.2.6.6.671 GLONASS_L2_CA_FREQ_HZ

```
double GLONASS_L2_CA_FREQ_HZ = FREQ2_GLO  [constexpr]
```

L2 [Hz].
Definition at line 68 of file GLONASS_L1_L2_CA.h.

10.2.6.6.672 GLONASS_L2_CA_SYMBOL_PERIOD_S

```
double GLONASS_L2_CA_SYMBOL_PERIOD_S = 0.01  [constexpr]
```

GLONASS L1 C/A symbol period [seconds].
Definition at line 73 of file GLONASS_L1_L2_CA.h.

10.2.6.6.673 GLONASS_L2_CA_SYMBOL_RATE_BPS

```
double GLONASS_L2_CA_SYMBOL_RATE_BPS = 1000.0  [constexpr]
```

Definition at line 75 of file GLONASS_L1_L2_CA.h.

10.2.6.6.674 GLONASS_LEAP_SECONDS

```
double GLONASS_LEAP_SECONDS[19][7]  [constexpr]
```

Initial value:

```
                                = {
{2017, 1, 1, 0, 0, 0, -18},
{2015, 7, 1, 0, 0, 0, -17},
{2012, 7, 1, 0, 0, 0, -16},
{2009, 1, 1, 0, 0, 0, -15},
{2006, 1, 1, 0, 0, 0, -14},
{1999, 1, 1, 0, 0, 0, -13},
{1997, 7, 1, 0, 0, 0, -12},
{1996, 1, 1, 0, 0, 0, -11},
{1994, 7, 1, 0, 0, 0, -10},
{1993, 7, 1, 0, 0, 0, -9},
{1992, 7, 1, 0, 0, 0, -8},
{1991, 1, 1, 0, 0, 0, -7},
{1990, 1, 1, 0, 0, 0, -6},
{1988, 1, 1, 0, 0, 0, -5},
{1985, 7, 1, 0, 0, 0, -4},
{1983, 7, 1, 0, 0, 0, -3},
{1982, 7, 1, 0, 0, 0, -2},
{1981, 7, 1, 0, 0, 0, -1},
{}}
```

Record of leap seconds definition for GLOT to GPST conversion and vice versa.
Each entry is defined by an array of 7 elements consisting of yr,month,day,hr,min,sec,utc-gpst

Note

Ideally should use leap seconds definitions of rtklib

Definition at line 124 of file GLONASS_L1_L2_CA.h.

10.2.6.6.675 GLONASS_MOON_ECCENTRICITY

```
double GLONASS_MOON_ECCENTRICITY = 0.054900489  [constexpr]
```

Eccentricity of lunar orbit.
Definition at line 58 of file GLONASS_L1_L2_CA.h.

10.2.6.6.676 GLONASS_MOON_GM

```
double GLONASS_MOON_GM = 4902.835  [constexpr]
```

Lunar gravitational constant [km^3/s^2].
Definition at line 56 of file GLONASS_L1_L2_CA.h.

10.2.6.6.677 GLONASS_MOON_INCLINATION

```
double GLONASS_MOON_INCLINATION = 0.000089803977407e3  [constexpr]
```
Inclination of lunar orbit to ecliptic plane (5 deg 08 min 43.4 sec) [rad].
Definition at line 59 of file GLONASS_L1_L2_CA.h.

10.2.6.6.678 GLONASS_MOON_OMEGA_0

```
double GLONASS_MOON_OMEGA_0 = 0.004523601514852e3  [constexpr]
```
(259 deg 10 min 59.79 sec) [rad]
Definition at line 54 of file GLONASS_L1_L2_CA.h.

10.2.6.6.679 GLONASS_MOON_OMEGA_1

```
double GLONASS_MOON_OMEGA_1 = -0.033757146246552e3  [constexpr]
```
(-1934 deg 08 min 31.23 sec) [rad]
Definition at line 55 of file GLONASS_L1_L2_CA.h.

10.2.6.6.680 GLONASS_MOON_Q0

```
double GLONASS_MOON_Q0 = -0.001115184961435e3  [constexpr]
```
(-63 deg 53 min 43.41 sec) [rad]
Definition at line 52 of file GLONASS_L1_L2_CA.h.

10.2.6.6.681 GLONASS_MOON_Q1

```
double GLONASS_MOON_Q1 = 8.328691103668023e3  [constexpr]
```
(477198 deg 50 min 56.79 sec) [rad]
Definition at line 53 of file GLONASS_L1_L2_CA.h.

10.2.6.6.682 GLONASS_MOON_SEMI_MAJOR_AXIS

```
double GLONASS_MOON_SEMI_MAJOR_AXIS = 3.84385243e5  [constexpr]
```
Semi-major axis of lunar orbit [km];.
Definition at line 57 of file GLONASS_L1_L2_CA.h.

10.2.6.6.683 GLONASS_OMEGA_EARTH_DOT

```
double GLONASS_OMEGA_EARTH_DOT = 7.292115e-5  [constexpr]
```
Earth rotation rate, [rad/s] ICD L1, L2 GLONASS Edition 5.1 2008 pag. 55.
Definition at line 39 of file MATH_CONSTANTS.h.

10.2.6.6.684 GLONASS_PRN

```
const std::map<uint32_t, int32_t> GLONASS_PRN
```
Definition at line 147 of file GLONASS_L1_L2_CA.h.

10.2.6.6.685 GLONASS_SEMI_MAJOR_AXIS

```
double GLONASS_SEMI_MAJOR_AXIS = 6378136  [constexpr]
```
Semi-major axis of Earth [m].
Definition at line 36 of file GLONASS_L1_L2_CA.h.

10.2.6.6.686 GLONASS_SUN_ECCENTRICITY

```
double GLONASS_SUN_ECCENTRICITY = 0.016719  [constexpr]
```
Eccentricity of solar orbit.
Definition at line 66 of file GLONASS_L1_L2_CA.h.

10.2.6.6.687 GLONASS_SUN_GM

```
double GLONASS_SUN_GM = 0.1325263e12  [constexpr]
```
Solar gravitational constant [km^3/s^2].
Definition at line 64 of file GLONASS_L1_L2_CA.h.

10.2.6.6.688 GLONASS_SUN_OMEGA

```
double GLONASS_SUN_OMEGA = 0.004908229466869e3  [constexpr]
```
TODO What is this operation in the seconds with T?(281 deg 13 min 15.0 + 6189.03 x T sec) [rad].
Definition at line 61 of file GLONASS_L1_L2_CA.h.

10.2.6.6.689 GLONASS_SUN_Q0

```
double GLONASS_SUN_Q0 = 0.006256583774423e3  [constexpr]
```
(358 deg 28 min 33.04 sec) [rad]
Definition at line 62 of file GLONASS_L1_L2_CA.h.

10.2.6.6.690 GLONASS_SUN_Q1

```
double GLONASS_SUN_Q1 = 0e3  [constexpr]
```
TODO Why is the value greater than 60?(129596579.10 sec) [rad].
Definition at line 63 of file GLONASS_L1_L2_CA.h.

10.2.6.6.691 GLONASS_SUN_SEMI_MAJOR_AXIS

```
double GLONASS_SUN_SEMI_MAJOR_AXIS = 1.49598e8  [constexpr]
```
Semi-major axis of solar orbit [km];.
Definition at line 65 of file GLONASS_L1_L2_CA.h.

10.2.6.6.692 GLONASS_TAU_0

```
double GLONASS_TAU_0 = -0.005835151531174e3  [constexpr]
```
(-334 deg 19 min 46.40 sec) [rad];
Definition at line 49 of file GLONASS_L1_L2_CA.h.

10.2.6.6.693 GLONASS_TAU_1

```
double GLONASS_TAU_1 = 0.071018041257371e3  [constexpr]
```
(4069 deg 02 min 02.52 sec) [rad];
Definition at line 50 of file GLONASS_L1_L2_CA.h.

10.2.6.6.694 GLONASS_U0

```
double GLONASS_U0 = 62636861.4  [constexpr]
```
Normal potential at surface of common terrestrial ellipsoid [m^2/s^2].
Definition at line 44 of file GLONASS_L1_L2_CA.h.

10.2.6.6.695 GNSS_OMEGA_EARTH_DOT

```
double GNSS_OMEGA_EARTH_DOT = 7.2921151467e-5  [constexpr]
```
Default Earth rotation rate, [rad/s].
Definition at line 26 of file MATH_CONSTANTS.h.

10.2.6.6.696 GNSS_PI

```
double GNSS_PI = 3.1415926535898  [constexpr]
```
pi constant as defined for GNSS
Definition at line 47 of file MATH_CONSTANTS.h.

10.2.6.6.697 GPS_CA_PREAMBLE

```
char GPS_CA_PREAMBLE[9] = "10001011"  [constexpr]
```
Definition at line 73 of file GPS_L1_CA.h.

10.2.6.6.698 GPS_CA_PREAMBLE_DURATION_MS

```
int32_t GPS_CA_PREAMBLE_DURATION_MS = 160  [constexpr]
```
Definition at line 62 of file GPS_L1_CA.h.

10.2.6.6.699 GPS_CA_PREAMBLE_DURATION_S

```
double GPS_CA_PREAMBLE_DURATION_S = 0.160  [constexpr]
```
Definition at line 59 of file GPS_L1_CA.h.

10.2.6.6.700 GPS_CA_PREAMBLE_LENGTH_BITS

```
int32_t GPS_CA_PREAMBLE_LENGTH_BITS = 8  [constexpr]
```
Definition at line 60 of file GPS_L1_CA.h.

10.2.6.6.701 GPS_CA_PREAMBLE_LENGTH_SYMBOLS

```
int32_t GPS_CA_PREAMBLE_LENGTH_SYMBOLS = 160  [constexpr]
```
Definition at line 61 of file GPS_L1_CA.h.

10.2.6.6.702 GPS_CA_PREAMBLE_SYMBOLS_STR

```
char GPS_CA_PREAMBLE_SYMBOLS_STR[161] = "1111111111111111111100000000000000000000000000000000000000000000000000000000
[constexpr]
```
Definition at line 74 of file GPS_L1_CA.h.

10.2.6.6.703 GPS_CA_TELEMETRY_RATE_BITS_SECOND

```
int32_t GPS_CA_TELEMETRY_RATE_BITS_SECOND = 50  [constexpr]
```
NAV message bit rate [bits/s].
Definition at line 63 of file GPS_L1_CA.h.

10.2.6.6.704 GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND

```
int32_t GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND = GPS_CA_TELEMETRY_RATE_BITS_SECOND * GPS_CA_↩
TELEMETRY_SYMBOLS_PER_BIT  [constexpr]
```
NAV message bit rate [symbols/s].
Definition at line 65 of file GPS_L1_CA.h.

10.2.6.6.705 GPS_CA_TELEMETRY_SYMBOLS_PER_BIT

```
int32_t GPS_CA_TELEMETRY_SYMBOLS_PER_BIT = 20  [constexpr]
```
Definition at line 64 of file GPS_L1_CA.h.

10.2.6.6.706 GPS_CNAV_DATA_PAGE_BITS

```
int32_t GPS_CNAV_DATA_PAGE_BITS = 300  [constexpr]
```
Definition at line 35 of file GPS_CNAV.h.

10.2.6.6.707 GPS_F

```
double GPS_F = -4.442807633e-10  [constexpr]
```
Constant, [s/(m)^(1/2)], IS-GPS-200M, 20.3.3.3.3.1.
Definition at line 32 of file MATH_CONSTANTS.h.

10.2.6.6.708 GPS_GM

```
double GPS_GM = 3.986005e14  [constexpr]
```
Universal gravitational constant times the mass of the Earth, [m^3/s^2] IS-GPS-200M, 20.3.3.3.3.1.
Definition at line 31 of file MATH_CONSTANTS.h.

10.2.6.6.709 GPS_L1_CA_BIT_PERIOD_MS

```
uint32_t GPS_L1_CA_BIT_PERIOD_MS = 20U  [constexpr]
```
GPS L1 C/A bit period [ms].
Definition at line 40 of file GPS_L1_CA.h.

10.2.6.6.710 GPS_L1_CA_CHIP_PERIOD_S

```
double GPS_L1_CA_CHIP_PERIOD_S = 9.7752e-07  [constexpr]
```
GPS L1 C/A chip period [seconds].
Definition at line 38 of file GPS_L1_CA.h.

10.2.6.6.711 GPS_L1_CA_CODE_LENGTH_CHIPS

```
double GPS_L1_CA_CODE_LENGTH_CHIPS = 1023.0  [constexpr]
```
GPS L1 C/A code length [chips].
Definition at line 36 of file GPS_L1_CA.h.

10.2.6.6.712 GPS_L1_CA_CODE_PERIOD_MS

```
uint32_t GPS_L1_CA_CODE_PERIOD_MS = 1U  [constexpr]
```
GPS L1 C/A code period [ms].
Definition at line 39 of file GPS_L1_CA.h.

10.2.6.6.713 GPS_L1_CA_CODE_PERIOD_S

```
double GPS_L1_CA_CODE_PERIOD_S = 0.001  [constexpr]
```
GPS L1 C/A code period [seconds].
Definition at line 37 of file GPS_L1_CA.h.

10.2.6.6.714 GPS_L1_CA_CODE_RATE_CPS

```
double GPS_L1_CA_CODE_RATE_CPS = 1.023e6  [constexpr]
```
GPS L1 C/A code rate [chips/s].
Definition at line 35 of file GPS_L1_CA.h.

10.2.6.6.715 GPS_L1_CA_HISTORY_DEEP

```
int32_t GPS_L1_CA_HISTORY_DEEP = 100  [constexpr]
```
Definition at line 56 of file GPS_L1_CA.h.

10.2.6.6.716 GPS_L1_CA_OPT_ACQ_FS_SPS

```
uint32_t GPS_L1_CA_OPT_ACQ_FS_SPS = 2000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 53 of file GPS_L1_CA.h.

10.2.6.6.717 GPS_L1_FREQ_HZ

```
double GPS_L1_FREQ_HZ = FREQ1  [constexpr]
```
L1 [Hz].
Definition at line 34 of file GPS_L1_CA.h.

10.2.6.6.718 GPS_L2_CNAV_DATA_PAGE_BITS

```
int32_t GPS_L2_CNAV_DATA_PAGE_BITS = 300  [constexpr]
```
GPS L2 CNAV page length, including preamble and CRC [bits].
Definition at line 45 of file GPS_L2C.h.

10.2.6.6.719 GPS_L2_CNAV_DATA_PAGE_DURATION_S

```
int32_t GPS_L2_CNAV_DATA_PAGE_DURATION_S = 12  [constexpr]
```
Definition at line 49 of file GPS_L2C.h.

10.2.6.6.720 GPS_L2_CNAV_DATA_PAGE_SYMBOLS

```
int32_t GPS_L2_CNAV_DATA_PAGE_SYMBOLS = 600  [constexpr]
```
Definition at line 48 of file GPS_L2C.h.

10.2.6.6.721 GPS_L2_FREQ_HZ

```
double GPS_L2_FREQ_HZ = FREQ2  [constexpr]
```
L2 [Hz].
Definition at line 35 of file GPS_L2C.h.

10.2.6.6.722 GPS_L2_L_CODE_LENGTH_CHIPS

```
int32_t GPS_L2_L_CODE_LENGTH_CHIPS = 767250  [constexpr]
```
GPS L2 L code length [chips].
Definition at line 44 of file GPS_L2C.h.

10.2.6.6.723 GPS_L2_L_CODE_RATE_CPS

```
double GPS_L2_L_CODE_RATE_CPS = 0.5115e6  [constexpr]
```
GPS L2 L code rate [chips/s].
Definition at line 40 of file GPS_L2C.h.

10.2.6.6.724 GPS_L2_L_PERIOD_S

```
double GPS_L2_L_PERIOD_S = 1.5  [constexpr]
```
GPS L2 L code period [seconds].
Definition at line 36 of file GPS_L2C.h.

10.2.6.6.725 GPS_L2_M_CODE_LENGTH_CHIPS

```
int32_t GPS_L2_M_CODE_LENGTH_CHIPS = 10230  [constexpr]
```
GPS L2 M code length [chips].
Definition at line 42 of file GPS_L2C.h.

10.2.6.6.726 GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED

```
int32_t GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED = 2 * GPS_L2_M_CODE_LENGTH_CHIPS  [constexpr]
```
GPS L2 M code length with L2 CL zeroed [chips].
Definition at line 43 of file GPS_L2C.h.

10.2.6.6.727 GPS_L2_M_CODE_PERIOD_MS

```
uint32_t GPS_L2_M_CODE_PERIOD_MS = 20  [constexpr]
```
GPS L2 L code period [ms].
Definition at line 41 of file GPS_L2C.h.

10.2.6.6.728 GPS_L2_M_CODE_RATE_CPS

```
double GPS_L2_M_CODE_RATE_CPS = 0.5115e6  [constexpr]
```

GPS L2 M code rate [chips/s].
Definition at line 37 of file GPS_L2C.h.

10.2.6.6.729 GPS_L2_M_CODE_RATE_L_ZEROED_CPS

```
double GPS_L2_M_CODE_RATE_L_ZEROED_CPS = 1.023e6  [constexpr]
```

GPS L2 M code rate with L2 M Zeroed [chips/s].
Definition at line 38 of file GPS_L2C.h.

10.2.6.6.730 GPS_L2_M_PERIOD_S

```
double GPS_L2_M_PERIOD_S = 0.02  [constexpr]
```

GPS L2 M code period [seconds].
Definition at line 39 of file GPS_L2C.h.

10.2.6.6.731 GPS_L2_SAMPLES_PER_SYMBOL

```
int32_t GPS_L2_SAMPLES_PER_SYMBOL = 1  [constexpr]
```

Definition at line 47 of file GPS_L2C.h.

10.2.6.6.732 GPS_L2_SYMBOLS_PER_BIT

```
int32_t GPS_L2_SYMBOLS_PER_BIT = 2  [constexpr]
```

Definition at line 46 of file GPS_L2C.h.

10.2.6.6.733 GPS_L2C_HISTORY_DEEP

```
int32_t GPS_L2C_HISTORY_DEEP = 5  [constexpr]
```

Definition at line 51 of file GPS_L2C.h.

10.2.6.6.734 GPS_L2C_M_INIT_REG

```
int32_t GPS_L2C_M_INIT_REG[115]  [constexpr]
```

Initial value:

```
                                =
{0742417664, 0756014035, 0002747144, 0066265724,
    0601403471, 0703232733, 0124510070, 0617316361,
    0047541621, 0733031046, 0713512145, 0024437606,
    0021264003, 0230655351, 0001314400, 0222021506,
    0540264026, 0205521705, 0064022144, 0120161274,
    0044023533, 0724744327, 0045743577, 0741201660,
    0700274134, 0010247261, 0713433445, 0737324162,
    0311627434, 0710452007, 0722462133, 0050172213,
    0500653703, 0755077436, 0136717361, 0756675453,
    0435506112, 0771353753, 0226107701, 0022025110,
    0402466344, 0752566114, 0702011164, 0041216771,
    0047457275, 0266333164, 0713167356, 0060546335,
    0355173035, 0617201036, 0157465571, 0767360553,
    0023127030, 0431343777, 0747317317, 0045706125,
    0002744276, 0060036467, 0217744147, 0603340174,
    0326616775, 0063240065, 0111460621,
    0604055104, 0157065232, 0013305707, 0603552017,
    0230461355, 0603653437, 0652346475, 0743107103,
    0401521277, 0167335110, 0014013575, 0362051132,
    0617753265, 0216363634, 0755561123, 0365304033,
    0625025543, 0054420334, 0415473671, 0662364360,
    0373446602, 0417564100, 0000526452, 0226631300,
    0113752074, 0706134401, 0041352546, 0664630154,
    0276524255, 0714720530, 0714051771, 0044526647,
    0207164322, 0262120161, 0204244652, 0202133131,
    0714351204, 0657127260, 0130567507, 0670517677,
    0607275514, 0045413633, 0212645405, 0613700455,
    0706202440, 0705056276, 0020373522, 0746013617,
    0132720621, 0434015513, 0566721727, 0140633660}
```

Definition at line 56 of file GPS_L2C.h.

10.2.6.6.735 GPS_L2C_OPT_ACQ_FS_SPS

```
uint32_t GPS_L2C_OPT_ACQ_FS_SPS = 2000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 54 of file GPS_L2C.h.

10.2.6.6.736 GPS_L5_CNAV_DATA_PAGE_BITS

```
int32_t GPS_L5_CNAV_DATA_PAGE_BITS = 300  [constexpr]
```
GPS L5 CNAV page length, including preamble and CRC [bits].
Definition at line 162 of file GPS_L5.h.

10.2.6.6.737 GPS_L5_CNAV_DATA_PAGE_DURATION_S

```
int32_t GPS_L5_CNAV_DATA_PAGE_DURATION_S = 6  [constexpr]
```
Definition at line 166 of file GPS_L5.h.

10.2.6.6.738 GPS_L5_CNAV_DATA_PAGE_SYMBOLS

```
int32_t GPS_L5_CNAV_DATA_PAGE_SYMBOLS = 600  [constexpr]
```
Definition at line 165 of file GPS_L5.h.

10.2.6.6.739 GPS_L5_FREQ_HZ

```
double GPS_L5_FREQ_HZ = FREQ5  [constexpr]
```
L5 [Hz].
Definition at line 32 of file GPS_L5.h.

10.2.6.6.740 GPS_L5_HISTORY_DEEP

```
int32_t GPS_L5_HISTORY_DEEP = 5  [constexpr]
```
Definition at line 43 of file GPS_L5.h.

10.2.6.6.741 GPS_L5_OPT_ACQ_FS_SPS

```
uint32_t GPS_L5_OPT_ACQ_FS_SPS = 10000000  [constexpr]
```
Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
Definition at line 46 of file GPS_L5.h.

10.2.6.6.742 GPS_L5_SAMPLES_PER_SYMBOL

```
int32_t GPS_L5_SAMPLES_PER_SYMBOL = 10  [constexpr]
```
Definition at line 164 of file GPS_L5.h.

10.2.6.6.743 GPS_L5_SYMBOLS_PER_BIT

```
int32_t GPS_L5_SYMBOLS_PER_BIT = 2  [constexpr]
```
Definition at line 163 of file GPS_L5.h.

10.2.6.6.744 GPS_L5I_CODE_LENGTH_CHIPS

```
int32_t GPS_L5I_CODE_LENGTH_CHIPS = 10230  [constexpr]
```
GPS L5I code length [chips].
Definition at line 39 of file GPS_L5.h.

10.2.6.6.745 GPS_L5I_CODE_RATE_CPS

```
double GPS_L5I_CODE_RATE_CPS = 10.23e6  [constexpr]
```
GPS L5I code rate [chips/s].
Definition at line 33 of file GPS_L5.h.

10.2.6.6.746 GPS_L5I_INIT_REG

```
int32_t GPS_L5I_INIT_REG[210]  [constexpr]
```
Definition at line 48 of file GPS_L5.h.

10.2.6.6.747 GPS_L5I_NH_CODE

```
int32_t GPS_L5I_NH_CODE[10] = {0, 0, 0, 0, 1, 1, 0, 1, 0, 1}  [constexpr]
```
Definition at line 168 of file GPS_L5.h.

10.2.6.6.748 GPS_L5I_NH_CODE_LENGTH

```
int32_t GPS_L5I_NH_CODE_LENGTH = 10  [constexpr]
```
Definition at line 167 of file GPS_L5.h.

10.2.6.6.749 GPS_L5I_NH_CODE_STR

```
char GPS_L5I_NH_CODE_STR[11] = "0000110101"  [constexpr]
```
Definition at line 171 of file GPS_L5.h.

10.2.6.6.750 GPS_L5I_PERIOD_MS

```
int32_t GPS_L5I_PERIOD_MS = 1  [constexpr]
```
GPS L5I code period [ms].
Definition at line 40 of file GPS_L5.h.

10.2.6.6.751 GPS_L5I_PERIOD_S

```
double GPS_L5I_PERIOD_S = 0.001  [constexpr]
```
GPS L5I code period [seconds].
Definition at line 34 of file GPS_L5.h.

10.2.6.6.752 GPS_L5I_SYMBOL_PERIOD_MS

```
int32_t GPS_L5I_SYMBOL_PERIOD_MS = 10  [constexpr]
```
GPS L5I symbol period [ms].
Definition at line 41 of file GPS_L5.h.

10.2.6.6.753 GPS_L5I_SYMBOL_PERIOD_S

```
double GPS_L5I_SYMBOL_PERIOD_S = 0.01  [constexpr]
```
GPS L5I symbol period [seconds].
Definition at line 35 of file GPS_L5.h.

10.2.6.6.754 GPS_L5Q_CODE_LENGTH_CHIPS

```
int32_t GPS_L5Q_CODE_LENGTH_CHIPS = 10230  [constexpr]
```
GPS L5Q code length [chips].
Definition at line 38 of file GPS_L5.h.

10.2.6.6.755 GPS_L5Q_CODE_RATE_CPS

```
double GPS_L5Q_CODE_RATE_CPS = 10.23e6  [constexpr]
```
GPS L5Q code rate [chips/s].
Definition at line 36 of file GPS_L5.h.

10.2.6.6.756 GPS_L5Q_INIT_REG

```
int32_t GPS_L5Q_INIT_REG[210]  [constexpr]
```
Definition at line 105 of file GPS_L5.h.

10.2.6.6.757 GPS_L5Q_NH_CODE

```
int32_t GPS_L5Q_NH_CODE[20] = {0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0}
[constexpr]
```

Definition at line 170 of file GPS_L5.h.

10.2.6.6.758 GPS_L5Q_NH_CODE_LENGTH

```
int32_t GPS_L5Q_NH_CODE_LENGTH = 20  [constexpr]
```

Definition at line 169 of file GPS_L5.h.

10.2.6.6.759 GPS_L5Q_NH_CODE_STR

```
char GPS_L5Q_NH_CODE_STR[21] = "00000100110101001110"  [constexpr]
```

Definition at line 172 of file GPS_L5.h.

10.2.6.6.760 GPS_L5Q_PERIOD_S

```
double GPS_L5Q_PERIOD_S = 0.001  [constexpr]
```

GPS L5Q code period [seconds].
Definition at line 37 of file GPS_L5.h.

10.2.6.6.761 GPS_LNAV_DATA_ID

```
int32_t GPS_LNAV_DATA_ID = 1  [constexpr]
```

IS-GPS-200N LNAV Data ID number 2, encoded as binary 01.
Definition at line 72 of file GPS_L1_CA.h.

10.2.6.6.762 GPS_SUBFRAME_BITS

```
int32_t GPS_SUBFRAME_BITS = 300  [constexpr]
```

Number of bits per subframe in the NAV message [bits].
Definition at line 68 of file GPS_L1_CA.h.

10.2.6.6.763 GPS_SUBFRAME_LENGTH

```
int32_t GPS_SUBFRAME_LENGTH = 40  [constexpr]
```

GPS_WORD_LENGTH x 10 = 40 bytes.
Definition at line 67 of file GPS_L1_CA.h.

10.2.6.6.764 GPS_SUBFRAME_MS

```
int32_t GPS_SUBFRAME_MS = 6000  [constexpr]
```

Subframe duration [seconds].
Definition at line 70 of file GPS_L1_CA.h.

10.2.6.6.765 GPS_SUBFRAME_SECONDS

```
int32_t GPS_SUBFRAME_SECONDS = 6  [constexpr]
```

Subframe duration [seconds].
Definition at line 69 of file GPS_L1_CA.h.

10.2.6.6.766 GPS_SV_CONFIG_AS_FLAG_MASK

```
int32_t GPS_SV_CONFIG_AS_FLAG_MASK = 0x8  [constexpr]
```

A-S flag (1 = ON).
Definition at line 190 of file GPS_L1_CA.h.

10.2.6.6.767 GPS_SV_CONFIG_BLOCK_IIF

```
int32_t GPS_SV_CONFIG_BLOCK_IIF = 3  [constexpr]
```

011: first with L5
Definition at line 194 of file GPS_L1_CA.h.

10.2.6.6.768 GPS_SV_CONFIG_BLOCK_IIR_M

```
int32_t GPS_SV_CONFIG_BLOCK_IIR_M = 2  [constexpr]
```

010: first with L2C
Definition at line 193 of file GPS_L1_CA.h.

10.2.6.6.769 GPS_SV_CONFIG_CODE_MASK

```
int32_t GPS_SV_CONFIG_CODE_MASK = 0x7  [constexpr]
```

SV configuration.
Definition at line 191 of file GPS_L1_CA.h.

10.2.6.6.770 GPS_SV_CONFIG_UNKNOWN

```
int32_t GPS_SV_CONFIG_UNKNOWN = 0  [constexpr]
```

000: no information
Definition at line 192 of file GPS_L1_CA.h.

10.2.6.6.771 GPS_WORD_BITS

```
int32_t GPS_WORD_BITS = 30  [constexpr]
```

Number of bits per word in the NAV message [bits].
Definition at line 71 of file GPS_L1_CA.h.

10.2.6.6.772 GPS_WORD_LENGTH

```
int32_t GPS_WORD_LENGTH = 4  [constexpr]
```

CRC + GPS WORD (-2 -1 0 ... 29) Bits = 4 bytes.
Definition at line 66 of file GPS_L1_CA.h.

10.2.6.6.773 HALF_PI

```
double HALF_PI = GNSS_PI / 2.0  [constexpr]
```

pi/2
Definition at line 48 of file MATH_CONSTANTS.h.

10.2.6.6.774 HAS_MSG_CODE_BIAS_LENGTH

```
size_t HAS_MSG_CODE_BIAS_LENGTH = 11  [constexpr]
```

Definition at line 54 of file Galileo_CNAV.h.

10.2.6.6.775 HAS_MSG_CODE_BIAS_SCALE_FACTOR

```
float HAS_MSG_CODE_BIAS_SCALE_FACTOR = 0.02  [constexpr]
```

Definition at line 97 of file Galileo_CNAV.h.

10.2.6.6.776 HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH

```
size_t HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH = 13  [constexpr]
```

Definition at line 49 of file Galileo_CNAV.h.

10.2.6.6.777 HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH

```
size_t HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH = 13  [constexpr]
```

Definition at line 53 of file Galileo_CNAV.h.

10.2.6.6.778 HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH

```
size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH = 2  [constexpr]
```
Definition at line 48 of file Galileo_CNAV.h.

10.2.6.6.779 HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH

```
size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH = 2  [constexpr]
```
Definition at line 52 of file Galileo_CNAV.h.

10.2.6.6.780 HAS_MSG_DELTA_CLOCK_SCALE_FACTOR

```
float HAS_MSG_DELTA_CLOCK_SCALE_FACTOR = 0.0025  [constexpr]
```
Definition at line 96 of file Galileo_CNAV.h.

10.2.6.6.781 HAS_MSG_DELTA_CROSS_TRACK_LENGTH

```
size_t HAS_MSG_DELTA_CROSS_TRACK_LENGTH = 12  [constexpr]
```
Definition at line 47 of file Galileo_CNAV.h.

10.2.6.6.782 HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR

```
float HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR = 0.008  [constexpr]
```
Definition at line 95 of file Galileo_CNAV.h.

10.2.6.6.783 HAS_MSG_DELTA_IN_TRACK_LENGTH

```
size_t HAS_MSG_DELTA_IN_TRACK_LENGTH = 12  [constexpr]
```
Definition at line 46 of file Galileo_CNAV.h.

10.2.6.6.784 HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR

```
float HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR = 0.008  [constexpr]
```
Definition at line 94 of file Galileo_CNAV.h.

10.2.6.6.785 HAS_MSG_DELTA_RADIAL_LENGTH

```
size_t HAS_MSG_DELTA_RADIAL_LENGTH = 13  [constexpr]
```
Definition at line 45 of file Galileo_CNAV.h.

10.2.6.6.786 HAS_MSG_DELTA_RADIAL_SCALE_FACTOR

```
float HAS_MSG_DELTA_RADIAL_SCALE_FACTOR = 0.0025  [constexpr]
```
Definition at line 93 of file Galileo_CNAV.h.

10.2.6.6.787 HAS_MSG_GALILEO_SYSTEM

```
uint8_t HAS_MSG_GALILEO_SYSTEM = 2  [constexpr]
```
Definition at line 103 of file Galileo_CNAV.h.

10.2.6.6.788 HAS_MSG_GPS_SYSTEM

```
uint8_t HAS_MSG_GPS_SYSTEM = 0  [constexpr]
```
Definition at line 102 of file Galileo_CNAV.h.

10.2.6.6.789 HAS_MSG_ID_CLOCK_SUBSET_LENGTH

```
size_t HAS_MSG_ID_CLOCK_SUBSET_LENGTH = 4  [constexpr]
```
Definition at line 51 of file Galileo_CNAV.h.

10.2.6.6.790 HAS_MSG_ID_MASK_LENGTH

```
size_t HAS_MSG_ID_MASK_LENGTH = 4  [constexpr]
```
Definition at line 37 of file Galileo_CNAV.h.

10.2.6.6.791 HAS_MSG_IOD_GAL_LENGTH

```
size_t HAS_MSG_IOD_GAL_LENGTH = 10  [constexpr]
```
Definition at line 44 of file Galileo_CNAV.h.

10.2.6.6.792 HAS_MSG_IOD_GPS_LENGTH

```
size_t HAS_MSG_IOD_GPS_LENGTH = 8  [constexpr]
```
Definition at line 43 of file Galileo_CNAV.h.

10.2.6.6.793 HAS_MSG_MASK_RESERVED_LENGTH

```
size_t HAS_MSG_MASK_RESERVED_LENGTH = 6  [constexpr]
```
Definition at line 41 of file Galileo_CNAV.h.

10.2.6.6.794 HAS_MSG_NAV_MESSAGE_LENGTH

```
size_t HAS_MSG_NAV_MESSAGE_LENGTH = 3  [constexpr]
```
Definition at line 40 of file Galileo_CNAV.h.

10.2.6.6.795 HAS_MSG_NSYS_LENGTH

```
size_t HAS_MSG_NSYS_LENGTH = 4  [constexpr]
```
Definition at line 36 of file Galileo_CNAV.h.

10.2.6.6.796 HAS_MSG_NSYS_SUB_LENGTH

```
size_t HAS_MSG_NSYS_SUB_LENGTH = 4  [constexpr]
```
Definition at line 50 of file Galileo_CNAV.h.

10.2.6.6.797 HAS_MSG_NUMBER_GNSS_IDS

```
int32_t HAS_MSG_NUMBER_GNSS_IDS = 16  [constexpr]
```
Definition at line 88 of file Galileo_CNAV.h.

10.2.6.6.798 HAS_MSG_NUMBER_MASK_IDS

```
int32_t HAS_MSG_NUMBER_MASK_IDS = 32  [constexpr]
```
Definition at line 87 of file Galileo_CNAV.h.

10.2.6.6.799 HAS_MSG_NUMBER_MAX_TOH

```
uint16_t HAS_MSG_NUMBER_MAX_TOH = 3599  [constexpr]
```
Definition at line 100 of file Galileo_CNAV.h.

10.2.6.6.800 HAS_MSG_NUMBER_MESSAGE_IDS

```
int32_t HAS_MSG_NUMBER_MESSAGE_IDS = 32  [constexpr]
```
Definition at line 89 of file Galileo_CNAV.h.

10.2.6.6.801 HAS_MSG_NUMBER_SATELLITE_IDS

```
int32_t HAS_MSG_NUMBER_SATELLITE_IDS = 40  [constexpr]
```
Definition at line 90 of file Galileo_CNAV.h.

10.2.6.6.802 HAS_MSG_NUMBER_SIGNAL_MASKS

```
int32_t HAS_MSG_NUMBER_SIGNAL_MASKS = 16  [constexpr]
```
Definition at line 91 of file Galileo_CNAV.h.

10.2.6.6.803 HAS_MSG_PHASE_BIAS_LENGTH

```
size_t HAS_MSG_PHASE_BIAS_LENGTH = 11  [constexpr]
```
Definition at line 55 of file Galileo_CNAV.h.

10.2.6.6.804 HAS_MSG_PHASE_BIAS_SCALE_FACTOR

```
float HAS_MSG_PHASE_BIAS_SCALE_FACTOR = 0.01  [constexpr]
```
Definition at line 98 of file Galileo_CNAV.h.

10.2.6.6.805 HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH

```
size_t HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH = 2  [constexpr]
```
Definition at line 56 of file Galileo_CNAV.h.

10.2.6.6.806 HAS_MSG_SATELLITE_MASK_LENGTH

```
size_t HAS_MSG_SATELLITE_MASK_LENGTH = 40  [constexpr]
```
Definition at line 38 of file Galileo_CNAV.h.

10.2.6.6.807 HAS_MSG_SIGNAL_MASK_LENGTH

```
size_t HAS_MSG_SIGNAL_MASK_LENGTH = 16  [constexpr]
```
Definition at line 39 of file Galileo_CNAV.h.

10.2.6.6.808 HAS_MSG_VALIDITY_INDEX_LENGTH

```
size_t HAS_MSG_VALIDITY_INDEX_LENGTH = 4  [constexpr]
```
Definition at line 42 of file Galileo_CNAV.h.

10.2.6.6.809 HAS_MSG_WRONG_SYSTEM

```
uint8_t HAS_MSG_WRONG_SYSTEM = 255  [constexpr]
```
Definition at line 104 of file Galileo_CNAV.h.

10.2.6.6.810 HAS_SIGNAL_INDEX_TABLE

```
const std::unordered_map<uint8_t, std::unordered_map<uint8_t, std::string> > HAS_SIGNAL_↩
INDEX_TABLE
```
Definition at line 129 of file Galileo_CNAV.h.

10.2.6.6.811 HAS_VALIDITY_INTERVALS

```
const std::unordered_map<uint8_t, uint16_t> HAS_VALIDITY_INTERVALS
```
Initial value:

```
                                                = {
{0, 5},
{1, 10},
{2, 15},
{3, 20},
{4, 30},
{5, 60},
{6, 90},
{7, 120},
{8, 180},
{9, 240},
{10, 300},
{11, 600},
{12, 900},
{13, 1800},
```

{14, 3600}}
Definition at line 168 of file Galileo_CNAV.h.

10.2.6.6.812 I_0_2_LSB

double I_0_2_LSB = PI_TWO_N31 [constexpr]
Definition at line 75 of file Galileo_INAV.h.

10.2.6.6.813 I_0_LSB

double I_0_LSB = PI_TWO_N31 [constexpr]
Definition at line 135 of file GPS_L1_CA.h.

10.2.6.6.814 I_DOT_2_LSB

double I_DOT_2_LSB = PI_TWO_N43 [constexpr]
Definition at line 79 of file Galileo_INAV.h.

10.2.6.6.815 I_DOT_LSB

double I_DOT_LSB = PI_TWO_N43 [constexpr]
Definition at line 144 of file GPS_L1_CA.h.

10.2.6.6.816 INAV_RS_BUFFER_LENGTH

size_t INAV_RS_BUFFER_LENGTH = 118 [constexpr]
Definition at line 275 of file Galileo_INAV.h.

10.2.6.6.817 INAV_RS_INFO_VECTOR_LENGTH

size_t INAV_RS_INFO_VECTOR_LENGTH = 58 [constexpr]
Definition at line 274 of file Galileo_INAV.h.

10.2.6.6.818 INAV_RS_PARITY_VECTOR_LENGTH

size_t INAV_RS_PARITY_VECTOR_LENGTH = 60 [constexpr]
Definition at line 273 of file Galileo_INAV.h.

10.2.6.6.819 INAV_RS_SUBVECTOR_LENGTH

size_t INAV_RS_SUBVECTOR_LENGTH = 15 [constexpr]
Definition at line 272 of file Galileo_INAV.h.

10.2.6.6.820 M0_10_LSB

double M0_10_LSB = TWO_N15 [constexpr]
Definition at line 237 of file Galileo_INAV.h.

10.2.6.6.821 M0_1_LSB

double M0_1_LSB = PI_TWO_N31 [constexpr]
Definition at line 62 of file Galileo_INAV.h.

10.2.6.6.822 M0_7_LSB

double M0_7_LSB = TWO_N15 [constexpr]
Definition at line 180 of file Galileo_INAV.h.

10.2.6.6.823 M0_9_LSB

```
double M0_9_LSB = TWO_N15  [constexpr]
```
Definition at line 212 of file Galileo_INAV.h.

10.2.6.6.824 M_0_LSB

```
double M_0_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 112 of file GPS_L1_CA.h.

10.2.6.6.825 MAX_SECONDS_REMEMBERING_MASK_IOD

```
uint64_t MAX_SECONDS_REMEMBERING_MASK_IOD = 1800  [constexpr]
```
Definition at line 59 of file Galileo_CNAV.h.

10.2.6.6.826 MAX_SECONDS_REMEMBERING_MID

```
uint64_t MAX_SECONDS_REMEMBERING_MID = 150  [constexpr]
```
Definition at line 58 of file Galileo_CNAV.h.

10.2.6.6.827 MAX_TOA_DELAY_MS

```
double MAX_TOA_DELAY_MS = 20.0  [constexpr]
```
Maximum Time-Of-Arrival (TOA) difference between satellites for a receiver operated on Earth surface is 20 ms. According to the GPS orbit model described in [1] Pag. 32. It should be taken into account to set the buffer size for the PRN start timestamp in the pseudoranges block. [1] J. Bao-Yen Tsui, Fundamentals of Global Positioning System Receivers. A Software Approach, John Wiley & Sons, Inc., Hoboken, NJ, 2nd edition, 2005.
Definition at line 50 of file GPS_L1_CA.h.

10.2.6.6.828 MAXCODE

```
int32_t MAXCODE = 57  [constexpr]
```
max number of obs code
Definition at line 88 of file gnss_obs_codes.h.

10.2.6.6.829 OMEGA0_10_LSB

```
double OMEGA0_10_LSB = TWO_N15  [constexpr]
```
Definition at line 233 of file Galileo_INAV.h.

10.2.6.6.830 OMEGA0_7_LSB

```
double OMEGA0_7_LSB = TWO_N15  [constexpr]
```
Definition at line 176 of file Galileo_INAV.h.

10.2.6.6.831 OMEGA0_8_LSB

```
double OMEGA0_8_LSB = TWO_N15  [constexpr]
```
Definition at line 201 of file Galileo_INAV.h.

10.2.6.6.832 OMEGA_0_2_LSB

```
double OMEGA_0_2_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 73 of file Galileo_INAV.h.

10.2.6.6.833 OMEGA_0_LSB

```
double OMEGA_0_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 131 of file GPS_L1_CA.h.

10.2.6.6.834 OMEGA_2_LSB

```
double OMEGA_2_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 77 of file Galileo_INAV.h.

10.2.6.6.835 OMEGA_7_LSB

```
double OMEGA_7_LSB = TWO_N15  [constexpr]
```
Definition at line 172 of file Galileo_INAV.h.

10.2.6.6.836 OMEGA_8_LSB

```
double OMEGA_8_LSB = TWO_N15  [constexpr]
```
Definition at line 197 of file Galileo_INAV.h.

10.2.6.6.837 OMEGA_9_LSB

```
double OMEGA_9_LSB = TWO_N15  [constexpr]
```
Definition at line 225 of file Galileo_INAV.h.

10.2.6.6.838 OMEGA_DOT_10_LSB

```
double OMEGA_DOT_10_LSB = TWO_N33  [constexpr]
```
Definition at line 235 of file Galileo_INAV.h.

10.2.6.6.839 OMEGA_DOT_3_LSB

```
double OMEGA_DOT_3_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 85 of file Galileo_INAV.h.

10.2.6.6.840 OMEGA_DOT_7_LSB

```
double OMEGA_DOT_7_LSB = TWO_N33  [constexpr]
```
Definition at line 178 of file Galileo_INAV.h.

10.2.6.6.841 OMEGA_DOT_8_LSB

```
double OMEGA_DOT_8_LSB = TWO_N33  [constexpr]
```
Definition at line 203 of file Galileo_INAV.h.

10.2.6.6.842 OMEGA_DOT_LSB

```
double OMEGA_DOT_LSB = PI_TWO_N43  [constexpr]
```
Definition at line 141 of file GPS_L1_CA.h.

10.2.6.6.843 OMEGA_LSB

```
double OMEGA_LSB = PI_TWO_N31  [constexpr]
```
Definition at line 139 of file GPS_L1_CA.h.

10.2.6.6.844 OSNMA_TABLE_1

```
const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_1
```
Initial value:

```
                                    = {
{0, std::string("Reserved")},
{1, std::string("Test")},
{2, std::string("Operational")},
{3, std::string("Don't use")}}
```
Definition at line 36 of file Galileo_OSNMA.h.

10.2.6.6.845 OSNMA_TABLE_10

```
const std::unordered_map<uint8_t, uint16_t> OSNMA_TABLE_10
```

Initial value:

```
                                    = {
{0, 96},
{1, 104},
{2, 112},
{3, 120},
{4, 128},
{5, 160},
{6, 192},
{7, 224},
{8, 256},
{9, 0},
{10, 0},
{11, 0},
{12, 0},
{13, 0},
{14, 0},
{15, 0}}
```

Definition at line 120 of file Galileo_OSNMA.h.

10.2.6.6.846 OSNMA_TABLE_11

```
const std::unordered_map<uint8_t, uint8_t> OSNMA_TABLE_11
```

Initial value:

```
                                       = {
    {0, 0},
    {1, 0},
    {2, 0},
    {3, 0},
    {4, 0},
    {5, 20},
    {6, 24},
    {7, 28},
    {8, 32},
    {9, 40},
    {10, 0},
    {11, 0},
    {12, 0},
    {13, 0},
    {14, 0},
    {15, 0},
}
```

Definition at line 138 of file Galileo_OSNMA.h.

10.2.6.6.847 OSNMA_TABLE_15

```
const std::unordered_map<std::string, uint16_t> OSNMA_TABLE_15
```

Initial value:

```
                                    = {
{std::string("ECDSA P-256"), 512},
{std::string("ECDSA P-521"), 1056}}
```

Definition at line 157 of file Galileo_OSNMA.h.

10.2.6.6.848 OSNMA_TABLE_16

```
const std::unordered_map<uint8_t, Mack_lookup> OSNMA_TABLE_16
```

Initial value:

```
                                    = {
{27, {2, 6, {"00S", "00E", "00E", "00E", "12S", "00E"}, {"00S ", "00E", "00E", "04S", "12S", "00E"}}},
{28, {2, 10, {"00S", "00E", "00E", "00E", "00S", "00E", "00E", "12S", "00E", "00E"}, {"00S", "00E", "00E",
      "00S", "00E", "00E", "04S", "12S", "00E", "00E"}}},
{31, {2, 5, {"00S", "00E", "00E", "12S", "00E"}, {"00S", "00E", "00E", "12S", "04S"}}},
{33, {2, 6, {"00S", "00E", "04S", "00E", "12S", "00E"}, {"00S", "00E", "00E", "12S", "00E", "12E"}}},
{34, {2, 6, {"00S", "FLX", "04S", "FLX", "12S", "00E"}, {"00S", "FLX", "00E", "12S", "00E", "12E"}}},
{35, {2, 6, {"00S", "FLX", "04S", "FLX", "12S", "FLX"}, {"00S", "FLX", "FLX", "12S", "FLX", "FLX"}}},
{36, {2, 5, {"00S", "FLX", "04S", "FLX", "12S"}, {"00S", "FLX", "00E", "12S", "12E"}}},
{37, {2, 5, {"00S", "00E", "04S", "00E", "12S"}, {"00S", "00E", "00E", "12S", "12E"}}},
{38, {2, 5, {"00S", "FLX", "04S", "FLX", "12S"}, {"00S", "FLX", "FLX", "12S", "FLX"}}},
{39, {2, 4, {"00S", "FLX", "04S", "FLX"}, {"00S", "FLX", "00E", "12S"}}},
{40, {2, 4, {"00S", "00E", "04S", "12S"}, {"00S", "00E", "00E", "12E"}}},
{41, {2, 4, {"00S", "FLX", "04S", "FLX"}, {"00S", "FLX", "FLX", "12S"}}}}
```

Definition at line 183 of file Galileo_OSNMA.h.

10.2.6.6.849 OSNMA_TABLE_2

```
const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_2
```

Initial value:

```
                                = {
{0, std::string("Reserved")},
{1, std::string("Nominal")},
{2, std::string("End of Chain (EOC)")},
{3, std::string("Chain Revoked (CREV)")},
{4, std::string("New Public Key (NPK)")},
{5, std::string("Public Key Revoked (PKREV)")},
{6, std::string("New Merkle Tree (NMT)")},
{7, std::string("Alert Message (AM)")}}
```

Definition at line 44 of file Galileo_OSNMA.h.

10.2.6.6.850 OSNMA_TABLE_3

```
const std::unordered_map<uint8_t, std::pair<uint16_t, uint16_t> > OSNMA_TABLE_3
```

Initial value:

```
                                = {
{0, {0, 0}},
{1, {0, 0}},
{2, {0, 0}},
{3, {0, 0}},
{4, {0, 0}},
{5, {0, 0}},
{6, {0, 0}},
{7, {13, 1352}},
{8, {14, 1456}},
{9, {15, 1560}},
{10, {16, 1664}},
{11, {0, 0}},
{12, {0, 0}},
{13, {0, 0}},
{14, {0, 0}},
{15, {0, 0}}}
```

Definition at line 55 of file Galileo_OSNMA.h.

10.2.6.6.851 OSNMA_TABLE_5

```
const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_5
```

Initial value:

```
                                = {
{0, std::string("Reserved")},
{1, std::string("ECDSA P-256")},
{2, std::string("Reserved")},
{3, std::string("ECDSA P-521")},
{4, std::string("OAM")},
{5, std::string("Reserved")},
{6, std::string("Reserved")},
{7, std::string("Reserved")},
{8, std::string("Reserved")},
{9, std::string("Reserved")},
{10, std::string("Reserved")},
{11, std::string("Reserved")},
{12, std::string("Reserved")},
{13, std::string("Reserved")},
{14, std::string("Reserved")},
{15, std::string("Reserved")}}
```

Definition at line 73 of file Galileo_OSNMA.h.

10.2.6.6.852 OSNMA_TABLE_6

```
const std::unordered_map<std::string, uint16_t> OSNMA_TABLE_6
```

Initial value:

```
                                = {
{std::string("ECDSA P-256"), 264},
{std::string("ECDSA P-521"), 536}}
```

Definition at line 91 of file Galileo_OSNMA.h.

10.2.6.6.853 OSNMA_TABLE_7

```
const std::unordered_map<uint8_t, std::pair<uint16_t, uint16_t> > OSNMA_TABLE_7
```

Initial value:

```
                                    = {
{0, {0, 0}},
{1, {7, 728}},
{2, {8, 832}},
{3, {9, 936}},
{4, {10, 1040}},
{5, {11, 1144}},
{6, {12, 1248}},
{7, {13, 1352}},
{8, {14, 1456}},
{9, {0, 0}},
{10, {0, 0}},
{11, {0, 0}},
{12, {0, 0}},
{13, {0, 0}},
{14, {0, 0}},
{15, {0, 0}}}
```

Definition at line 96 of file Galileo_OSNMA.h.

10.2.6.6.854 OSNMA_TABLE_8

```
const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_8
```

Initial value:

```
                                    = {
{0, std::string("SHA-256")},
{1, std::string("Reserved")},
{2, std::string("SHA3-256")},
{3, std::string("Reserved")}}
```

Definition at line 114 of file Galileo_OSNMA.h.

10.2.6.6.855 PI_TWO_N19

```
double PI_TWO_N19 = 5.992112452678286e-006  [constexpr]
```

Pi*2^-19.
Definition at line 117 of file MATH_CONSTANTS.h.

10.2.6.6.856 PI_TWO_N23

```
double PI_TWO_N23 = 3.745070282923929e-007  [constexpr]
```

Pi*2^-23.
Definition at line 121 of file MATH_CONSTANTS.h.

10.2.6.6.857 PI_TWO_N31

```
double PI_TWO_N31 = 1.462918079267160e-009  [constexpr]
```

Pi*2^-31.
Definition at line 119 of file MATH_CONSTANTS.h.

10.2.6.6.858 PI_TWO_N38

```
double PI_TWO_N38 = 1.142904749427469e-011  [constexpr]
```

Pi*2^-38.
Definition at line 120 of file MATH_CONSTANTS.h.

10.2.6.6.859 PI_TWO_N43

```
double PI_TWO_N43 = 3.571577341960839e-013  [constexpr]
```

Pi*2^-43.
Definition at line 118 of file MATH_CONSTANTS.h.

10.2.6.6.860 QZSS_CA_PREAMBLE_SYMBOLS_STR

```
const char QZSS_CA_PREAMBLE_SYMBOLS_STR[161] = "11111111111111111110000000000000000000000000000000000000000000000000
[constexpr]
```

Definition at line 55 of file qzss.h.

10.2.6.6.861 QZSS_L5I_NH_CODE_STR

```
const char QZSS_L5I_NH_CODE_STR[11] = "0000110101"  [constexpr]
```
Definition at line 57 of file qzss.h.

10.2.6.6.862 QZSS_L5Q_NH_CODE_STR

```
const char QZSS_L5Q_NH_CODE_STR[21] = "00000100110101001110"  [constexpr]
```
Definition at line 56 of file qzss.h.

10.2.6.6.863 R2D

```
double R2D = 180.0 / GNSS_PI  [constexpr]
```
rad to deg
Definition at line 124 of file MATH_CONSTANTS.h.

10.2.6.6.864 SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS

```
int32_t SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS = 2  [constexpr]
```
Encoded channel symbol period [ms] (1/500 sps).
Definition at line 52 of file SBAS_L1.h.

10.2.6.6.865 SBAS_L1_CHANNEL_SYMBOL_RATE_SPS

```
int32_t SBAS_L1_CHANNEL_SYMBOL_RATE_SPS = 500  [constexpr]
```
Encoded channel symbol rate (rate-1/2 FEC) [symbols/s].
Definition at line 51 of file SBAS_L1.h.

10.2.6.6.866 SBAS_L1_CODE_LENGTH_CHIPS

```
double SBAS_L1_CODE_LENGTH_CHIPS = 1023.0  [constexpr]
```
SBAS L1 C/A code length [chips].
Definition at line 36 of file SBAS_L1.h.

10.2.6.6.867 SBAS_L1_CODE_PERIOD_MS

```
uint32_t SBAS_L1_CODE_PERIOD_MS = 1U  [constexpr]
```
SBAS L1 C/A code period [ms].
Definition at line 38 of file SBAS_L1.h.

10.2.6.6.868 SBAS_L1_CODE_PERIOD_S

```
double SBAS_L1_CODE_PERIOD_S = 0.001  [constexpr]
```
SBAS L1 C/A code period [s].
Definition at line 37 of file SBAS_L1.h.

10.2.6.6.869 SBAS_L1_CODE_RATE_CPS

```
double SBAS_L1_CODE_RATE_CPS = 1.023e6  [constexpr]
```
SBAS L1 C/A code chip rate [chips/s].
Definition at line 35 of file SBAS_L1.h.

10.2.6.6.870 SBAS_L1_DATA_FIELD_BYTES

```
int32_t SBAS_L1_DATA_FIELD_BYTES = 29  [constexpr]
```
Bytes holding preamble, type, and 212-bit data field.
Definition at line 55 of file SBAS_L1.h.

10.2.6.6.871 SBAS_L1_DATA_RATE_BPS

```
int32_t SBAS_L1_DATA_RATE_BPS = 250  [constexpr]
```
Raw data bit rate [bits/s].
Definition at line 50 of file SBAS_L1.h.

10.2.6.6.872 SBAS_L1_FREQ_HZ

```
double SBAS_L1_FREQ_HZ = FREQ1  [constexpr]
```
SBAS L1 carrier frequency [Hz] (same as GPS L1).
Definition at line 34 of file SBAS_L1.h.

10.2.6.6.873 SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS

```
int32_t SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS = 1  [constexpr]
```
Definition at line 62 of file SBAS_L1.h.

10.2.6.6.874 SBAS_L1_MSG_LENGTH_BITS

```
int32_t SBAS_L1_MSG_LENGTH_BITS = 250  [constexpr]
```
Information bits per 1-second message frame [bits].
Definition at line 53 of file SBAS_L1.h.

10.2.6.6.875 SBAS_L1_MSG_LENGTH_BYTES

```
int32_t SBAS_L1_MSG_LENGTH_BYTES = 32  [constexpr]
```
Bytes holding a 250-bit frame with six trailing zero bits.
Definition at line 54 of file SBAS_L1.h.

10.2.6.6.876 SBAS_L1_OPT_ACQ_FS_SPS

```
uint32_t SBAS_L1_OPT_ACQ_FS_SPS = 2000000U  [constexpr]
```
Sampling frequency that maximises acquisition SNR [samples/s].
Definition at line 41 of file SBAS_L1.h.

10.2.6.6.877 SBAS_L1_PREAMBLE_1

```
uint8_t SBAS_L1_PREAMBLE_1 = 0x53U  [constexpr]
```
First preamble byte (binary: 0101 0011).
Definition at line 65 of file SBAS_L1.h.

10.2.6.6.878 SBAS_L1_PREAMBLE_2

```
uint8_t SBAS_L1_PREAMBLE_2 = 0x9AU  [constexpr]
```
Second preamble byte (binary: 1001 1010).
Definition at line 66 of file SBAS_L1.h.

10.2.6.6.879 SBAS_L1_PREAMBLE_3

```
uint8_t SBAS_L1_PREAMBLE_3 = 0xC6U  [constexpr]
```
Third preamble byte (binary: 1100 0110).
Definition at line 67 of file SBAS_L1.h.

10.2.6.6.880 SBAS_L1_PRN_MAX

```
uint32_t SBAS_L1_PRN_MAX = 138U  [constexpr]
```
Maximum SBAS PRN (inclusive).
Definition at line 45 of file SBAS_L1.h.

10.2.6.6.881 SBAS_L1_PRN_MIN

```
uint32_t SBAS_L1_PRN_MIN = 120U  [constexpr]
```

Minimum SBAS PRN (inclusive).
Definition at line 44 of file SBAS_L1.h.

10.2.6.6.882 SC2RAD

```
double SC2RAD = GNSS_PI  [constexpr]
```

semi-circle to radian (IS-GPS)
Definition at line 125 of file MATH_CONSTANTS.h.

10.2.6.6.883 SIGNAL_FREQ_MAP

```
const std::unordered_map<std::string, double> SIGNAL_FREQ_MAP
```

Initial value:

```
                                        = {
    {"1C", FREQ1},
    {"2S", FREQ2},
    {"L5", FREQ5},
    {"1B", FREQ1},
    {"5X", FREQ5},
    {"E6", FREQ6},
    {"7X", FREQ7},
    {"1G", FREQ1_GLO},
    {"2G", FREQ2_GLO},
    {"B1", FREQ1_BDS},
    {"B2", FREQ2_BDS},
    {"5D", FREQ5},
    {"B3", FREQ3_BDS},
    {"1D", FREQ1},
    {"J1", FREQ1},
    {"J5", FREQ5},
}
```

Definition at line 47 of file gnss_frequencies.h.

10.2.6.6.884 SIZE_DSM_BLOCKS_BYTES

```
size_t SIZE_DSM_BLOCKS_BYTES = 13  [constexpr]
```

Definition at line 33 of file Galileo_OSNMA.h.

10.2.6.6.885 SPEED_OF_LIGHT_M_MS

```
double SPEED_OF_LIGHT_M_MS = 299792.4580  [constexpr]
```

Speed of light in vacuum [m/ms].
Definition at line 28 of file MATH_CONSTANTS.h.

10.2.6.6.886 SPEED_OF_LIGHT_M_S

```
double SPEED_OF_LIGHT_M_S = 299792458.0  [constexpr]
```

Speed of light in vacuum [m/s].
Definition at line 27 of file MATH_CONSTANTS.h.

10.2.6.6.887 SQRT_A_LSB

```
double SQRT_A_LSB = TWO_N19  [constexpr]
```

Definition at line 120 of file GPS_L1_CA.h.

10.2.6.6.888 T0A_7_LSB

```
int32_t T0A_7_LSB = 600  [constexpr]
```

Definition at line 165 of file Galileo_INAV.h.

10.2.6.6.889 T0A_9_LSB

```
int32_t T0A_9_LSB = 600  [constexpr]
```
Definition at line 210 of file Galileo_INAV.h.

10.2.6.6.890 T0C_4_LSB

```
int32_t T0C_4_LSB = 60  [constexpr]
```
Definition at line 107 of file Galileo_INAV.h.

10.2.6.6.891 T0E_1_LSB

```
int32_t T0E_1_LSB = 60  [constexpr]
```
Definition at line 60 of file Galileo_INAV.h.

10.2.6.6.892 T0T_6_LSB

```
int32_t T0T_6_LSB = 3600  [constexpr]
```
Definition at line 153 of file Galileo_INAV.h.

10.2.6.6.893 T_0_G_10_LSB

```
int32_t T_0_G_10_LSB = 3600  [constexpr]
```
Definition at line 249 of file Galileo_INAV.h.

10.2.6.6.894 T_GD_LSB

```
double T_GD_LSB = TWO_N31  [constexpr]
```
Definition at line 94 of file GPS_L1_CA.h.

10.2.6.6.895 T_OA_LSB

```
int32_t T_OA_LSB = TWO_P12  [constexpr]
```
Definition at line 244 of file GPS_L1_CA.h.

10.2.6.6.896 T_OC_LSB

```
int32_t T_OC_LSB = static_cast<int32_t>(TWO_P4)  [constexpr]
```
Definition at line 97 of file GPS_L1_CA.h.

10.2.6.6.897 T_OE_LSB

```
int32_t T_OE_LSB = static_cast<int32_t>(TWO_P4)  [constexpr]
```
Definition at line 122 of file GPS_L1_CA.h.

10.2.6.6.898 T_OT_LSB

```
double T_OT_LSB = TWO_P12  [constexpr]
```
Definition at line 174 of file GPS_L1_CA.h.

10.2.6.6.899 TEXT_BLACK

```
const std::string TEXT_BLACK = "\033[30m"
```
Definition at line 36 of file display.h.

10.2.6.6.900 TEXT_BLUE

```
const std::string TEXT_BLUE = "\033[34m"
```
Definition at line 40 of file display.h.

10.2.6.6.901 TEXT_BOLD_BLACK

```
const std::string TEXT_BOLD_BLACK = "\033[1m\033[30m"
```
Definition at line 44 of file display.h.

10.2.6.6.902 TEXT_BOLD_BLUE

```
const std::string TEXT_BOLD_BLUE = "\033[1m\033[34m"
```
Definition at line 48 of file display.h.

10.2.6.6.903 TEXT_BOLD_CYAN

```
const std::string TEXT_BOLD_CYAN = "\033[1m\033[36m"
```
Definition at line 50 of file display.h.

10.2.6.6.904 TEXT_BOLD_GREEN

```
const std::string TEXT_BOLD_GREEN = "\033[1m\033[32m"
```
Definition at line 46 of file display.h.

10.2.6.6.905 TEXT_BOLD_MAGENTA

```
const std::string TEXT_BOLD_MAGENTA = "\033[1m\033[35m"
```
Definition at line 49 of file display.h.

10.2.6.6.906 TEXT_BOLD_RED

```
const std::string TEXT_BOLD_RED = "\033[1m\033[31m"
```
Definition at line 45 of file display.h.

10.2.6.6.907 TEXT_BOLD_WHITE

```
const std::string TEXT_BOLD_WHITE = "\033[1m\033[37m"
```
Definition at line 51 of file display.h.

10.2.6.6.908 TEXT_BOLD_YELLOW

```
const std::string TEXT_BOLD_YELLOW = "\033[1m\033[33m"
```
Definition at line 47 of file display.h.

10.2.6.6.909 TEXT_CYAN

```
const std::string TEXT_CYAN = "\033[36m"
```
Definition at line 42 of file display.h.

10.2.6.6.910 TEXT_GREEN

```
const std::string TEXT_GREEN = "\033[32m"
```
Definition at line 38 of file display.h.

10.2.6.6.911 TEXT_MAGENTA

```
const std::string TEXT_MAGENTA = "\033[35m"
```
Definition at line 41 of file display.h.

10.2.6.6.912 TEXT_RED

```
const std::string TEXT_RED = "\033[31m"
```
Definition at line 37 of file display.h.

10.2.6.6.913 TEXT_RESET

```
const std::string TEXT_RESET = "\033[0m"
```
Definition at line 35 of file display.h.

10.2.6.6.914 TEXT_WHITE

```
const std::string TEXT_WHITE = "\033[37m"
```
Definition at line 43 of file display.h.

10.2.6.6.915 TEXT_YELLOW

```
const std::string TEXT_YELLOW = "\033[33m"
```
Definition at line 39 of file display.h.

10.2.6.6.916 TWO_N10

```
double TWO_N10 = 0.0009765625  [constexpr]
```
2^-10
Definition at line 77 of file MATH_CONSTANTS.h.

10.2.6.6.917 TWO_N11

```
double TWO_N11 = 4.882812500000000e-004  [constexpr]
```
2^-11
Definition at line 78 of file MATH_CONSTANTS.h.

10.2.6.6.918 TWO_N14

```
double TWO_N14 = 0.00006103515625  [constexpr]
```
2^-14
Definition at line 79 of file MATH_CONSTANTS.h.

10.2.6.6.919 TWO_N15

```
double TWO_N15 = 3.051757812500000e-005  [constexpr]
```
2^-15
Definition at line 80 of file MATH_CONSTANTS.h.

10.2.6.6.920 TWO_N16

```
double TWO_N16 = 1.525878906250000e-005  [constexpr]
```
2^-16
Definition at line 81 of file MATH_CONSTANTS.h.

10.2.6.6.921 TWO_N17

```
double TWO_N17 = 7.629394531250000e-006  [constexpr]
```
2^-17
Definition at line 82 of file MATH_CONSTANTS.h.

10.2.6.6.922 TWO_N18

```
double TWO_N18 = 3.814697265625000e-006  [constexpr]
```
2^-18
Definition at line 83 of file MATH_CONSTANTS.h.

10.2.6.6.923 TWO_N19

```
double TWO_N19 = 1.907348632812500e-006  [constexpr]
```

2^-19
Definition at line 84 of file MATH_CONSTANTS.h.

10.2.6.6.924 TWO_N2

```
double TWO_N2 = 0.25  [constexpr]
```

2^-2
Definition at line 72 of file MATH_CONSTANTS.h.

10.2.6.6.925 TWO_N20

```
double TWO_N20 = 9.536743164062500e-007  [constexpr]
```

2^-20
Definition at line 85 of file MATH_CONSTANTS.h.

10.2.6.6.926 TWO_N21

```
double TWO_N21 = 4.768371582031250e-007  [constexpr]
```

2^-21
Definition at line 86 of file MATH_CONSTANTS.h.

10.2.6.6.927 TWO_N22

```
double TWO_N22 = 2.384185791015625e-007  [constexpr]
```

2^-22
Definition at line 87 of file MATH_CONSTANTS.h.

10.2.6.6.928 TWO_N23

```
double TWO_N23 = 1.192092895507810e-007  [constexpr]
```

2^-23
Definition at line 88 of file MATH_CONSTANTS.h.

10.2.6.6.929 TWO_N24

```
double TWO_N24 = 5.960464477539063e-008  [constexpr]
```

2^-24
Definition at line 89 of file MATH_CONSTANTS.h.

10.2.6.6.930 TWO_N25

```
double TWO_N25 = 2.980232238769531e-008  [constexpr]
```

2^-25
Definition at line 90 of file MATH_CONSTANTS.h.

10.2.6.6.931 TWO_N26

```
double TWO_N26 = 1.490116119384765e-009  [constexpr]
```

2^-26
Definition at line 91 of file MATH_CONSTANTS.h.

10.2.6.6.932 TWO_N27

```
double TWO_N27 = 7.450580596923828e-009  [constexpr]
```

2^-27
Definition at line 92 of file MATH_CONSTANTS.h.

10.2.6.6.933 TWO_N29

```
double TWO_N29 = 1.862645149230957e-009 [constexpr]
```

2^-29
Definition at line 93 of file MATH_CONSTANTS.h.

10.2.6.6.934 TWO_N30

```
double TWO_N30 = 9.313225746154785e-010 [constexpr]
```

2^-30
Definition at line 94 of file MATH_CONSTANTS.h.

10.2.6.6.935 TWO_N31

```
double TWO_N31 = 4.656612873077393e-010 [constexpr]
```

2^-31
Definition at line 95 of file MATH_CONSTANTS.h.

10.2.6.6.936 TWO_N32

```
double TWO_N32 = 2.328306436538696e-010 [constexpr]
```

2^-32
Definition at line 96 of file MATH_CONSTANTS.h.

10.2.6.6.937 TWO_N33

```
double TWO_N33 = 1.164153218269348e-010 [constexpr]
```

2^-33
Definition at line 97 of file MATH_CONSTANTS.h.

10.2.6.6.938 TWO_N34

```
double TWO_N34 = 5.82076609134674e-011 [constexpr]
```

2^-34
Definition at line 98 of file MATH_CONSTANTS.h.

10.2.6.6.939 TWO_N35

```
double TWO_N35 = 2.91038304567337e-011 [constexpr]
```

2^-35
Definition at line 99 of file MATH_CONSTANTS.h.

10.2.6.6.940 TWO_N38

```
double TWO_N38 = 3.637978807091713e-012 [constexpr]
```

2^-38
Definition at line 100 of file MATH_CONSTANTS.h.

10.2.6.6.941 TWO_N39

```
double TWO_N39 = 1.818989403545856e-012 [constexpr]
```

2^-39
Definition at line 101 of file MATH_CONSTANTS.h.

10.2.6.6.942 TWO_N40

```
double TWO_N40 = 9.094947017729280e-013 [constexpr]
```

2^-40
Definition at line 102 of file MATH_CONSTANTS.h.

10.2.6.6.943 TWO_N43

```
double TWO_N43 = 1.136868377216160e-013  [constexpr]
```

2^-43
Definition at line 103 of file MATH_CONSTANTS.h.

10.2.6.6.944 TWO_N44

```
double TWO_N44 = 5.684341886080802e-14  [constexpr]
```

2^-44
Definition at line 104 of file MATH_CONSTANTS.h.

10.2.6.6.945 TWO_N46

```
double TWO_N46 = 1.4210854715202e-014  [constexpr]
```

2^-46
Definition at line 105 of file MATH_CONSTANTS.h.

10.2.6.6.946 TWO_N48

```
double TWO_N48 = 3.552713678800501e-15  [constexpr]
```

2^-46
Definition at line 106 of file MATH_CONSTANTS.h.

10.2.6.6.947 TWO_N5

```
double TWO_N5 = 0.03125  [constexpr]
```

2^-5
Definition at line 73 of file MATH_CONSTANTS.h.

10.2.6.6.948 TWO_N50

```
double TWO_N50 = 8.881784197001252e-016  [constexpr]
```

2^-50
Definition at line 108 of file MATH_CONSTANTS.h.

10.2.6.6.949 TWO_N51

```
double TWO_N51 = 4.44089209850063e-016  [constexpr]
```

2^-51
Definition at line 109 of file MATH_CONSTANTS.h.

10.2.6.6.950 TWO_N55

```
double TWO_N55 = 2.775557561562891e-017  [constexpr]
```

2^-55
Definition at line 110 of file MATH_CONSTANTS.h.

10.2.6.6.951 TWO_N57

```
double TWO_N57 = 6.938893903907228e-18  [constexpr]
```

2^-57
Definition at line 111 of file MATH_CONSTANTS.h.

10.2.6.6.952 TWO_N59

```
double TWO_N59 = 1.73472347597681e-018  [constexpr]
```

2^-59
Definition at line 112 of file MATH_CONSTANTS.h.

10.2.6.6.953 TWO_N6

```
double TWO_N6 = 0.015625  [constexpr]
```
2^-6
Definition at line 74 of file MATH_CONSTANTS.h.

10.2.6.6.954 TWO_N60

```
double TWO_N60 = 8.673617379884036e-19  [constexpr]
```
2^-60
Definition at line 113 of file MATH_CONSTANTS.h.

10.2.6.6.955 TWO_N66

```
double TWO_N66 = 1.3552527156068805425093160010874271392822265625e-20  [constexpr]
```
2^-66
Definition at line 114 of file MATH_CONSTANTS.h.

10.2.6.6.956 TWO_N68

```
double TWO_N68 = 3.388131789017201e-21  [constexpr]
```
2^-68
Definition at line 115 of file MATH_CONSTANTS.h.

10.2.6.6.957 TWO_N8

```
double TWO_N8 = 0.00390625  [constexpr]
```
2^-8
Definition at line 75 of file MATH_CONSTANTS.h.

10.2.6.6.958 TWO_N9

```
double TWO_N9 = 0.001953125  [constexpr]
```
2^-9
Definition at line 76 of file MATH_CONSTANTS.h.

10.2.6.6.959 TWO_P11

```
double TWO_P11 = 2048.0  [constexpr]
```
2^11
Definition at line 62 of file MATH_CONSTANTS.h.

10.2.6.6.960 TWO_P12

```
double TWO_P12 = 4096.0  [constexpr]
```
2^12
Definition at line 63 of file MATH_CONSTANTS.h.

10.2.6.6.961 TWO_P14

```
double TWO_P14 = 16384.0  [constexpr]
```
2^14
Definition at line 64 of file MATH_CONSTANTS.h.

10.2.6.6.962 TWO_P16

```
double TWO_P16 = 65536.0  [constexpr]
```
2^16
Definition at line 65 of file MATH_CONSTANTS.h.

10.2.6.6.963 TWO_P19

```
double TWO_P19 = 524288.0  [constexpr]
```
2^19
Definition at line 66 of file MATH_CONSTANTS.h.

10.2.6.6.964 TWO_P3

```
double TWO_P3 = 8.0  [constexpr]
```
2^3
Definition at line 59 of file MATH_CONSTANTS.h.

10.2.6.6.965 TWO_P31

```
double TWO_P31 = 2147483648.0  [constexpr]
```
2^31
Definition at line 67 of file MATH_CONSTANTS.h.

10.2.6.6.966 TWO_P32

```
double TWO_P32 = 4294967296.0  [constexpr]
```
2^32
Definition at line 68 of file MATH_CONSTANTS.h.

10.2.6.6.967 TWO_P4

```
double TWO_P4 = 16.0  [constexpr]
```
2^4
Definition at line 60 of file MATH_CONSTANTS.h.

10.2.6.6.968 TWO_P56

```
double TWO_P56 = 7.205759403792794e+016  [constexpr]
```
2^56
Definition at line 69 of file MATH_CONSTANTS.h.

10.2.6.6.969 TWO_P57

```
double TWO_P57 = 1.441151880758559e+017  [constexpr]
```
2^57
Definition at line 70 of file MATH_CONSTANTS.h.

10.2.6.6.970 TWO_P8

```
double TWO_P8 = 256.0  [constexpr]
```
2^8
Definition at line 61 of file MATH_CONSTANTS.h.

10.2.6.6.971 TWO_PI

```
double TWO_PI = 2.0 * GNSS_PI  [constexpr]
```
2 * pi
Definition at line 49 of file MATH_CONSTANTS.h.

10.2.6.6.972 WN_LSF_LSB

```
double WN_LSF_LSB = 1  [constexpr]
```
Definition at line 180 of file GPS_L1_CA.h.

10.2.6.6.973 WN_T_LSB

```
double WN_T_LSB = 1  [constexpr]
```
Definition at line 176 of file GPS_L1_CA.h.

10.3 Channel

Topics

- channel_adapters
- channel_libs

10.3.1 Detailed Description

Classes containing a GNSS channel.

10.3.2 channel_adapters

Classes

- class Channel

  This class represents a GNSS channel. It wraps an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction through a Finite State Machine.

10.3.2.1 Detailed Description

Classes that wrap an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction.

10.3.3 channel_libs

Classes

- class ChannelFsm

  This class implements a State Machine for channel.
- class channel_msg_receiver_cc

  GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

Typedefs

- using channel_msg_receiver_cc_sptr = gnss_shared_ptr<channel_msg_receiver_cc>

Functions

- channel_msg_receiver_cc_sptr **channel_msg_receiver_make_cc** (std::shared_ptr< ChannelFsm > channel_fsm, bool repeat)

10.3.3.1 Detailed Description

Library with utilities for a GNSS Channel.

10.3.3.2 Typedef Documentation

10.3.3.2.1 channel_msg_receiver_cc_sptr

```
using channel_msg_receiver_cc_sptr = gnss_shared_ptr<channel_msg_receiver_cc>
```
Definition at line 33 of file channel_msg_receiver_cc.h.

10.4 Signal Conditioner

Topics

- conditioner_adapters

10.4.1 Detailed Description

Signal Conditioner wrapper block

10.4.2 conditioner_adapters

Classes

- class ArraySignalConditioner

  This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.
- class SignalConditioner

  This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.

10.4.2.1 Detailed Description

Wrap a Signal Conditioner with a GNSSBlockInterface

10.5 Data Type Adapters

Topics

- data_type_adapters
- data_type_gr_blocks

10.5.1 Detailed Description

Classes for data type conversion
Classes for data type conversion

10.5.2 data_type_adapters

Classes

- class ByteToShort

  Adapts an 8-bits sample stream (IF) to a short int stream (IF).
- class CshortToGrComplex

  Adapts an 16-bits complex sample stream to a float complex stream.
- class IbyteToCbyte
- class IbyteToComplex

  Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.
- class IbyteToCshort

  Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.
- class IshortToComplex

  Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.
- class IshortToCshort

  Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.

10.5.2.1 Detailed Description

Wrap GNU Radio data type adapter blocks with a GNSSBlockInterface
Wrap GNU Radio data type adapter blocks with a GNSSBlockInterface

10.5.3 data_type_gr_blocks

Classes

- class cshort_to_gr_complex

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<float> stream.
- class interleaved_byte_to_complex_byte

  This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::complex<unsigned char>).
- class interleaved_byte_to_complex_short

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.
- class interleaved_short_to_complex_short

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.

Typedefs

- using cshort_to_gr_complex_sptr = gnss_shared_ptr<cshort_to_gr_complex>
- using interleaved_byte_to_complex_byte_sptr = gnss_shared_ptr<interleaved_byte_to_complex_byte>
- using interleaved_byte_to_complex_short_sptr = gnss_shared_ptr<interleaved_byte_to_complex_short>
- using interleaved_short_to_complex_short_sptr = gnss_shared_ptr<interleaved_short_to_complex_short>

Functions

- cshort_to_gr_complex_sptr **make_cshort_to_gr_complex** ()
- interleaved_byte_to_complex_byte_sptr **make_interleaved_byte_to_complex_byte** ()
- interleaved_byte_to_complex_short_sptr **make_interleaved_byte_to_complex_short** ()
- interleaved_short_to_complex_short_sptr **make_interleaved_short_to_complex_short** ()

10.5.3.1 Detailed Description

GNU Radio Blocks for data type conversion

10.5.3.2 Typedef Documentation

10.5.3.2.1 cshort_to_gr_complex_sptr

```
using cshort_to_gr_complex_sptr = gnss_shared_ptr<cshort_to_gr_complex>
```
Definition at line 32 of file cshort_to_gr_complex.h.

10.5.3.2.2 interleaved_byte_to_complex_byte_sptr

```
using interleaved_byte_to_complex_byte_sptr = gnss_shared_ptr<interleaved_byte_to_complex_byte>
```
Definition at line 32 of file interleaved_byte_to_complex_byte.h.

10.5.3.2.3 interleaved_byte_to_complex_short_sptr

```
using interleaved_byte_to_complex_short_sptr = gnss_shared_ptr<interleaved_byte_to_complex_short>
```
Definition at line 32 of file interleaved_byte_to_complex_short.h.

10.5.3.2.4 interleaved_short_to_complex_short_sptr

```
using interleaved_short_to_complex_short_sptr = gnss_shared_ptr<interleaved_short_to_complex_short>
```
Definition at line 31 of file interleaved_short_to_complex_short.h.

10.6 Input Filter

Topics

- input_filter_adapters
- input_filter_gr_blocks

10.6.1 Detailed Description

Classes for input signal filtering

10.6.2 input_filter_adapters

Classes

- class BeamformerFilter

  Interface of an adapter of a digital beamformer block to a GNSSBlockInterface.
- class FirFilter

  This class adapts a GNU Radio gr_fir_filter designed with pm_remez.
- class FreqXlatingFirFilter

  This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez.
- class NotchFilter
- class NotchFilterLite
- class PulseBlankingFilter

10.6.2.1 Detailed Description

Classes that wrap GNU Radio input filters with a GNSSBlockInterface

10.6.3 input_filter_gr_blocks

Classes

- class beamformer

  This class implements a real-time software-defined spatial filter using the CTTC GNSS experimental antenna array input and a set of dynamically reloadable weights.
- class Notch

  This class implements a real-time software-defined multi state notch filter.
- class NotchLite

  This class implements a real-time software-defined multi state notch filter light version.
- class pulse_blanking_cc

Typedefs

- using beamformer_sptr = gnss_shared_ptr<beamformer>
- using notch_sptr = gnss_shared_ptr<Notch>
- using notch_lite_sptr = gnss_shared_ptr<NotchLite>
- using pulse_blanking_cc_sptr = gnss_shared_ptr<pulse_blanking_cc>

Functions

- beamformer_sptr **make_beamformer_sptr** ()
- notch_sptr **make_notch_filter** (float pfa, float p_c_factor, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)
- notch_lite_sptr **make_notch_filter_lite** (float p_c_factor, float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset, int32_t n_segments_coeff)
- pulse_blanking_cc_sptr **make_pulse_blanking_cc** (float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)

Variables

- const int GNSS_SDR_BEAMFORMER_CHANNELS = 8

10.6.3.1 Detailed Description

GNU Radio blocks implementing input filters,

10.6.3.2 Typedef Documentation

10.6.3.2.1 beamformer_sptr

```
using beamformer_sptr = gnss_shared_ptr<beamformer>
```
Definition at line 33 of file beamformer.h.

10.6.3.2.2 notch_lite_sptr

```
using notch_lite_sptr = gnss_shared_ptr<NotchLite>
```
Definition at line 36 of file notch_lite_cc.h.

10.6.3.2.3 notch_sptr

```
using notch_sptr = gnss_shared_ptr<Notch>
```
Definition at line 36 of file notch_cc.h.

10.6.3.2.4 pulse_blanking_cc_sptr

```
using pulse_blanking_cc_sptr = gnss_shared_ptr<pulse_blanking_cc>
```
Definition at line 36 of file pulse_blanking_cc.h.

10.6.3.3 Variable Documentation

10.6.3.3.1 GNSS_SDR_BEAMFORMER_CHANNELS

```
const int GNSS_SDR_BEAMFORMER_CHANNELS = 8
```
Definition at line 37 of file beamformer.h.

10.7 Algorithms Common Library

Topics

- algorithms_libs
- gnss_sdr_flags

10.7.1 Detailed Description

Common utilities for the GNSS receiver.

10.7.2 algorithms_libs

Namespaces

- namespace std

  STL namespace.

Classes

- class byte_x2_to_complex_byte

  This class adapts two signed char streams into a std::complex<signed char> stream.
- class complex_byte_to_float_x2

  This class adapts a std::complex<signed char> stream into two 16-bits (short) streams.
- class complex_float_to_complex_byte

  This class adapts a gr_complex stream into a std::complex<signed char> stream.
- class conjugate_cc

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.
- class conjugate_ic

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.
- class conjugate_sc

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.
- class cshort_to_float_x2

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.
- class Gnss_circular_deque< T >
- class Pass_Through

  This class implements a block that connects input and output (does nothing).
- struct SensorDataSample< DataType >
- class SensorDataAggregator
- class SensorDataFile
- class SensorDataSource
- struct SensorDataFileConfiguration
- struct SensorDataConfiguration
- class SensorDataSourceConfiguration
- struct SensorDataType
- struct SensorIdentifier
- class short_x2_to_cshort

  This class adapts two short streams into a std::complex<short> stream.
- class TrackingCmd

Typedefs

- using byte_x2_to_complex_byte_sptr = gnss_shared_ptr<byte_x2_to_complex_byte>
- using complex_byte_to_float_x2_sptr = gnss_shared_ptr<complex_byte_to_float_x2>
- using complex_float_to_complex_byte_sptr = gnss_shared_ptr<complex_float_to_complex_byte>
- using conjugate_cc_sptr = gnss_shared_ptr<conjugate_cc>
- using conjugate_ic_sptr = gnss_shared_ptr<conjugate_ic>
- using conjugate_sc_sptr = gnss_shared_ptr<conjugate_sc>
- using cshort_to_float_x2_sptr = gnss_shared_ptr<cshort_to_float_x2>
- using item_type_converter_t = std::function<void(void *, const void *, uint32_t)>
- using short_x2_to_cshort_sptr = gnss_shared_ptr<short_x2_to_cshort>

Functions

- void **beidou_b1c_code_gen_int** (own::span< int32_t > dest, const std::array< char, 3 > &signal_id, int32_t prn)
- void **beidou_b1c_code_gen_sinboc11_float** (own::span< float > dest, const std::array< char, 3 > &signal_id, uint32_t prn)
- void **beidou_b1c_code_gen_float_sampled** (own::span< float > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)
- void **beidou_b1c_code_gen_complex_sampled** (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)

- void **beidou_b1c_code_gen_complex_sampled** (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)
- void beidou_b1i_code_gen_int (own::span< int32_t > dest, int32_t prn, uint32_t chip_shift)

  Generates int32_t GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
- void **beidou_b2a_data_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_pilot_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_float** (own::span< float > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_complex** (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_complex_sampled** (own::span< std::complex< float > > dest, uint32_t prn, int sampling_freq, uint32_t chip_shift)
- void beidou_b3i_code_gen_int (own::span< int > dest, int32_t prn, uint32_t chip_shift)

  Generates int BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip_shift)

  Generates complex BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int sampling_freq, uint32_t chip_shift)

  Generates complex BeiDou B3I code for the desired SV ID and code shift, and sampled to specific sampling frequency.
- byte_x2_to_complex_byte_sptr **make_byte_x2_to_complex_byte** ()
- complex_byte_to_float_x2_sptr **make_complex_byte_to_float_x2** ()
- complex_float_to_complex_byte_sptr **make_complex_float_to_complex_byte** ()
- conjugate_cc_sptr **make_conjugate_cc** ()
- conjugate_ic_sptr **make_conjugate_ic** ()
- conjugate_sc_sptr **make_conjugate_sc** ()
- cshort_to_float_x2_sptr **make_cshort_to_float_x2** ()
- void galileo_e1_code_gen_sinboc11_float (own::span< float > dest, const std::array< char, 3 > &signal_id, uint32_t prn)

  This function generates Galileo E1 code (can select E1B or E1C sinboc).
- void galileo_e1_code_gen_float_sampled (own::span< float > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)

  This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).
- void galileo_e1_code_gen_float_sampled (own::span< float > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).
- void galileo_e1_code_gen_complex_sampled (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)

  This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).

- void galileo_e1_code_gen_complex_sampled (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  galileo_e1_code_gen_complex_sampled without secondary_flag for backward compatibility.
- void galileo_e5_a_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn, const std::array< char, 3 > &signal_id)

  Generates Galileo E5a code at 1 sample/chip.
- void galileo_e5_a_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, const std::array< char, 3 > &signal_id, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E5a complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e5_b_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn, const std::array< char, 3 > &signal_id)

  Generates Galileo E5b code at 1 sample/chip.
- void galileo_e5_b_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, const std::array< char, 3 > &signal_id, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E5b complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e6_b_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6B code at 1 sample/chip.
- void galileo_e6_b_code_gen_float_primary (own::span< float > dest, int32_t prn)

  Generates Galileo E6B code at 1 sample/chip.
- void galileo_e6_b_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E6B complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e6_c_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6C codes at 1 sample/chip.
- void galileo_e6_c_code_gen_float_primary (own::span< float > dest, int32_t prn)

  Generates Galileo E6C codes at 1 sample/chip.
- void galileo_e6_c_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E6C complex codes, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e6_c_secondary_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6C secondary codes at 1 sample/chip.
- void galileo_e6_c_secondary_code_gen_float (own::span< float > dest, int32_t prn)

  Generates Galileo E6C secondary codes at 1 sample/chip.
- std::string galileo_e6_c_secondary_code (int32_t prn)

  Generates a string with Galileo E6C secondary codes at 1 sample/chip.
- arma::mat Skew_symmetric (const arma::vec &a)

  Calculates skew-symmetric matrix.
- double **WGS84_g0** (double Lat_rad)
- double **WGS84_geocentric_radius** (double Lat_geodetic_rad)
- int topocent (double *Az, double *El, double *D, const arma::vec &x, const arma::vec &dx)

  Transformation of vector dx into topocentric coordinate system with origin at x Inputs: x - vector origin coordinates (in ECEF system [X; Y; Z;]) dx - vector ([dX; dY; dZ;]).
- int togeod (double *dphi, double *dlambda, double *h, double a, double finv, double X, double Y, double Z)

  Subroutine to calculate geodetic coordinates latitude, longitude, height given Cartesian coordinates X,Y,Z, and reference ellipsoid values semi-major axis (a) and the inverse of flattening (finv).
- arma::vec Gravity_ECEF (const arma::vec &r_eb_e)

  Calculates acceleration due to gravity resolved about ECEF-frame.
- arma::vec cart2geo (const arma::vec &XYZ, int elipsoid_selection)

  Conversion of Cartesian coordinates (X,Y,Z) to geographical coordinates (latitude, longitude, h) on a selected reference ellipsoid.
- arma::vec **LLH_to_deg** (const arma::vec &LLH)
- double **degtorad** (double angleInDegrees)

- double **radtodeg** (double angleInRadians)
- double **mstoknotsh** (double MetersPerSeconds)
- double **mstokph** (double MetersPerSeconds)
- arma::vec **CTM_to_Euler** (const arma::mat &C)
- arma::mat **Euler_to_CTM** (const arma::vec &eul)
- void **ECEF_to_Geo** (const arma::vec &r_eb_e, const arma::vec &v_eb_e, const arma::mat &C_b_e, arma::↩ vec &LLH, arma::vec &v_eb_n, arma::mat &C_b_n)
- void Geo_to_ECEF (const arma::vec &LLH, const arma::vec &v_eb_n, const arma::mat &C_b_n, arma::vec &r_eb_e, arma::vec &v_eb_e, arma::mat &C_b_e)

  From Geographic to ECEF coordinates.
- void pv_Geo_to_ECEF (double L_b, double lambda_b, double h_b, const arma::vec &v_eb_n, arma::vec &r_eb_e, arma::vec &v_eb_e)

  Converts curvilinear to Cartesian position and velocity resolving axes from NED to ECEF This function created 11/↩ 4/2012 by Paul Groves.
- double great_circle_distance (double lat1, double lon1, double lat2, double lon2)

  The Haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes.
- void cart2utm (const arma::vec &r_eb_e, int zone, arma::vec &r_enu)

  Transformation of ECEF (X,Y,Z) to (E,N,U) in UTM, zone 'zone'.
- int findUtmZone (double latitude_deg, double longitude_deg)

  Function finds the UTM zone number for given longitude and latitude.
- double clsin (const arma::colvec &ar, int degree, double argument)

  Clenshaw summation of sinus of argument.
- void clksin (const arma::colvec &ar, int degree, double arg_real, double arg_imag, double ∗re, double ∗im)

  Clenshaw summation of sinus with complex argument.
- void glonass_l1_ca_code_gen_complex (own::span< std::complex< float > > dest, uint32_t chip_shift)

  Generates complex GLONASS L1 C/A code for the desired SV ID and code shift.
- void glonass_l1_ca_code_gen_float (own::span< float > dest, uint32_t chip_shift)

  Generates float GLONASS L1 C/A code for the desired SV ID and code shift.
- void glonass_l1_ca_code_gen_complex_sampled (own::span< std::complex< float > > dest, int32_↩ t sampling_freq, uint32_t chip_shift)

  Generates complex GLONASS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
- void glonass_l2_ca_code_gen_complex (own::span< std::complex< float > > dest, uint32_t chip_shift)

  Generates complex GLONASS L2 C/A code for the desired SV ID and code shift.
- void glonass_l2_ca_code_gen_float (own::span< float > dest, uint32_t chip_shift)

  Generates float GLONASS L2 C/A code for the desired SV ID and code shift.
- void glonass_l2_ca_code_gen_complex_sampled (own::span< std::complex< float > > dest, int32_↩ t sampling_freq, uint32_t chip_shift)

  Generates complex GLONASS L2 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
- bool **gnss_sdr_create_directory** (const std::string &foldername)
- void complex_exp_gen (own::span< std::complex< float > > dest, double freq, double sampling_freq)

  This function generates a complex exponential in dest.
- void complex_exp_gen_conj (own::span< std::complex< float > > dest, double freq, double sampling_freq)

  This function generates a conjugate complex exponential in dest.
- void hex_to_binary_converter (own::span< int32_t > dest, char from)

  This function makes a conversion from hex (the input is a char) to binary (the output are 4 ints with +1 or -1 values).
- std::string hex_to_binary_string (char from)

  This function makes a conversion from hex (the input is a char) to binary (the output is a string of 4 char with 0 or 1 values).
- void resampler (const own::span< float > from, own::span< float > dest, float fs_in, float fs_out)

  This function resamples a sequence of float values.

- void resampler (own::span< const std::complex< float > > from, own::span< std::complex< float > > dest, float fs_in, float fs_out)

  This function resamples a sequence of complex values.
- void gps_l2c_m_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L2C M code for the desired SV ID.
- void gps_l2c_m_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates float GPS L2C M code for the desired SV ID.
- void gps_l2c_m_code_gen_float_cl_zeroed (own::span< float > dest, uint32_t prn)

  Generates float GPS L2C M code with double chip rate and L component zeroed for the desired SV ID.
- void gps_l2c_m_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency.
- void gps_l2c_m_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, bool cl_zeroed)

  Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency, optionally zeroing CL slots.
- void gps_l5i_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L5I code for the desired SV ID.
- void gps_l5i_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real GPS L5I code for the desired SV ID.
- void gps_l5q_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L5Q code for the desired SV ID.
- void gps_l5q_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real GPS L5Q code for the desired SV ID.
- void gps_l5i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L5I code for the desired SV ID, and sampled to specific sampling frequency.
- void gps_l5q_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L5Q code for the desired SV ID, and sampled to specific sampling frequency.
- void gps_l1_ca_code_gen_int (own::span< int32_t > dest, int32_t prn, uint32_t chip_shift)

  Generates int GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
- bool item_type_valid (const std::string &item_type)

  Check if a string is a valid item type.
- size_t item_type_size (const std::string &item_type)

  Return the size of the given item type, or zero if unknown.
- bool item_type_is_complex (const std::string &item_type)

  Determine if an item_type is complex.
- item_type_converter_t make_vector_converter (const std::string &input_type, const std::string &output_type)

  Create a function to convert an array of input_type to an array of output_type.
- bool qzss_l1_code_is_boc (uint32_t prn)

  Returns true if the given QZSS PRN broadcasts the BOC(1,1)-modulated L1 C/B signal (IS-QZSS-PNT-006).

- void qzss_l1_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L1 C/A(B) code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l1_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L1 C/A(B) code for the desired SV ID (QZSS_L1_SAMPLES_PER_CHIP samples per chip).
- void qzss_l5i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L5I code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l5i_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L5I code for the desired SV ID.
- void qzss_l5q_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L5Q code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l5q_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L5I code for the desired SV ID.
- void sbas_l1_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex SBAS L1 C/A code for the desired SV PRN, sampled at sampling_freq [Hz].
- void sbas_l1_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real SBAS L1 C/A code for the desired SV PRN (chip_shift=0).
- void sbas_l1_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn, uint32_t chip_shift)

  Generates complex SBAS L1 C/A code for the desired SV PRN with a given chip_shift.
- std::vector< gr::tag_t > resample_sensor_data_tags (const std::vector< gr::tag_t > &tags, double freq_in, double freq_out)

  Updates timestamp within sensor data tags. To be used in resampler blocks.
- short_x2_to_cshort_sptr **make_short_x2_to_cshort** ()
- Gnss_circular_deque< T >::Gnss_circular_deque ()

  Default constructor.
- Gnss_circular_deque< T >::Gnss_circular_deque (unsigned int max_size, unsigned int nchann)

  nchann = number of channels; max_size = channel capacity
- unsigned int Gnss_circular_deque< T >::size (unsigned int ch) const

  Returns the number of available elements in a channel.
- T & Gnss_circular_deque< T >::back (unsigned int ch)

  Returns a reference to the last element in the deque.
- T & Gnss_circular_deque< T >::front (unsigned int ch)

  Returns a reference to the first element in the deque.
- T & Gnss_circular_deque< T >::at (unsigned int ch, unsigned int pos)

  Returns a reference to an element with bound checking.
- const T & Gnss_circular_deque< T >::get (unsigned int ch, unsigned int pos) const

  Returns a const reference to an element without bound checking.
- void Gnss_circular_deque< T >::clear (unsigned int ch)

  Removes all the elements of the deque (Sets size to 0). Capacity is not modified.
- void Gnss_circular_deque< T >::reset (unsigned int max_size, unsigned int nchann)

  Removes all the elements in all the channels. Re-sets the number of channels and their capacity.
- void Gnss_circular_deque< T >::reset ()

  Removes all the channels (Sets nchann to 0).
- void Gnss_circular_deque< T >::pop_front (unsigned int ch)

  Removes the first element of the deque.
- void Gnss_circular_deque< T >::push_back (unsigned int ch, const T &new_data)

  Inserts an element at the end of the deque.

10.7.2.1 Detailed Description

Common utilities for GNSS algorithms.

10.7.2.2 Typedef Documentation

10.7.2.2.1 byte_x2_to_complex_byte_sptr

```
using byte_x2_to_complex_byte_sptr = gnss_shared_ptr<byte_x2_to_complex_byte>
```
Definition at line 33 of file byte_x2_to_complex_byte.h.

10.7.2.2.2 complex_byte_to_float_x2_sptr

```
using complex_byte_to_float_x2_sptr = gnss_shared_ptr<complex_byte_to_float_x2>
```
Definition at line 33 of file complex_byte_to_float_x2.h.

10.7.2.2.3 complex_float_to_complex_byte_sptr

```
using complex_float_to_complex_byte_sptr = gnss_shared_ptr<complex_float_to_complex_byte>
```
Definition at line 33 of file complex_float_to_complex_byte.h.

10.7.2.2.4 conjugate_cc_sptr

```
using conjugate_cc_sptr = gnss_shared_ptr<conjugate_cc>
```
Definition at line 33 of file conjugate_cc.h.

10.7.2.2.5 conjugate_ic_sptr

```
using conjugate_ic_sptr = gnss_shared_ptr<conjugate_ic>
```
Definition at line 33 of file conjugate_ic.h.

10.7.2.2.6 conjugate_sc_sptr

```
using conjugate_sc_sptr = gnss_shared_ptr<conjugate_sc>
```
Definition at line 33 of file conjugate_sc.h.

10.7.2.2.7 cshort_to_float_x2_sptr

```
using cshort_to_float_x2_sptr = gnss_shared_ptr<cshort_to_float_x2>
```
Definition at line 33 of file cshort_to_float_x2.h.

10.7.2.2.8 item_type_converter_t

```
using item_type_converter_t = std::function<void(void *, const void *, uint32_t)>
```
Definition at line 34 of file item_type_helpers.h.

10.7.2.2.9 short_x2_to_cshort_sptr

```
using short_x2_to_cshort_sptr = gnss_shared_ptr<short_x2_to_cshort>
```
Definition at line 33 of file short_x2_to_cshort.h.

10.7.2.3 Function Documentation

10.7.2.3.1 at()

```
template<class T>
T & Gnss_circular_deque< T >::at (
            unsigned int ch,
            unsigned int pos)
```
Returns a reference to an element with bound checking.
Definition at line 88 of file gnss_circular_deque.h.

10.7.2.3.2 back()

```
template<class T>
T & Gnss_circular_deque< T >::back (
            unsigned int ch)
```

Returns a reference to the last element in the deque.
Definition at line 74 of file gnss_circular_deque.h.

10.7.2.3.3 beidou_b1i_code_gen_complex()

```
void beidou_b1i_code_gen_complex (
            own::span< std::complex< float > > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates complex GPS L1 C/A code for the desired SV ID and code shift.
References beidou_b1i_code_gen_complex().
Referenced by beidou_b1i_code_gen_complex().

10.7.2.3.4 beidou_b1i_code_gen_complex_sampled()

```
void beidou_b1i_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
References beidou_b1i_code_gen_complex_sampled().
Referenced by beidou_b1i_code_gen_complex_sampled().

10.7.2.3.5 beidou_b1i_code_gen_float()

```
void beidou_b1i_code_gen_float (
            own::span< float > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates float GPS L1 C/A code for the desired SV ID and code shift.
References beidou_b1i_code_gen_float().
Referenced by beidou_b1i_code_gen_float().

10.7.2.3.6 beidou_b1i_code_gen_int()

```
void beidou_b1i_code_gen_int (
            own::span< int32_t > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates int32_t GPS L1 C/A code for the desired SV ID and code shift.
References beidou_b1i_code_gen_int().
Referenced by beidou_b1i_code_gen_int().

10.7.2.3.7 beidou_b3i_code_gen_complex()

```
void beidou_b3i_code_gen_complex (
            own::span< std::complex< float > > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates complex BeiDou B3I code for the desired SV ID and code shift.
References beidou_b3i_code_gen_complex().
Referenced by beidou_b3i_code_gen_complex().

10.7.2.3.8 beidou_b3i_code_gen_complex_sampled()

```
void beidou_b3i_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int sampling_freq,
            uint32_t chip_shift)
```

Generates complex BeiDou B3I code for the desired SV ID and code shift, and sampled to specific sampling frequency.
References beidou_b3i_code_gen_complex_sampled().
Referenced by beidou_b3i_code_gen_complex_sampled().

10.7.2.3.9 beidou_b3i_code_gen_float()

```
void beidou_b3i_code_gen_float (
            own::span< float > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates float BeiDou B3I code for the desired SV ID and code shift.
References beidou_b3i_code_gen_float().
Referenced by beidou_b3i_code_gen_float().

10.7.2.3.10 beidou_b3i_code_gen_int()

```
void beidou_b3i_code_gen_int (
            own::span< int > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates int BeiDou B3I code for the desired SV ID and code shift.
References beidou_b3i_code_gen_int().
Referenced by beidou_b3i_code_gen_int().

10.7.2.3.11 cart2geo()

```
arma::vec cart2geo (
            const arma::vec & XYZ,
            int elipsoid_selection)
```

Conversion of Cartesian coordinates (X,Y,Z) to geographical coordinates (latitude, longitude, h) on a selected reference ellipsoid.
Choices of Reference Ellipsoid for Geographical Coordinates

1. International Ellipsoid 1924
2. International Ellipsoid 1967
3. World Geodetic System 1972
4. Geodetic Reference System 1980
5. World Geodetic System 1984

References cart2geo().
Referenced by cart2geo().

10.7.2.3.12 cart2utm()

```
void cart2utm (
            const arma::vec & r_eb_e,
            int zone,
            arma::vec & r_enu)
```

Transformation of ECEF (X,Y,Z) to (E,N,U) in UTM, zone 'zone'.
References cart2utm().
Referenced by cart2utm().

10.7.2.3.13 clear()

```
template<class T>
void Gnss_circular_deque< T >::clear (
            unsigned int ch)
```

Removes all the elements of the deque (Sets size to 0). Capacity is not modified.
Definition at line 102 of file gnss_circular_deque.h.

10.7.2.3.14 clksin()

```
void clksin (
            const arma::colvec & ar,
            int degree,
            double arg_real,
            double arg_imag,
            double * re,
            double * im)
```

Clenshaw summation of sinus with complex argument.
References clksin().
Referenced by clksin().

10.7.2.3.15 clsin()

```
double clsin (
            const arma::colvec & ar,
            int degree,
            double argument)
```

Clenshaw summation of sinus of argument.
References clsin().
Referenced by clsin().

10.7.2.3.16 complex_exp_gen()

```
void complex_exp_gen (
            own::span< std::complex< float > > dest,
            double freq,
            double sampling_freq)
```

This function generates a complex exponential in dest.
References complex_exp_gen().
Referenced by complex_exp_gen().

10.7.2.3.17 complex_exp_gen_conj()

```
void complex_exp_gen_conj (
            own::span< std::complex< float > > dest,
            double freq,
            double sampling_freq)
```

This function generates a conjugate complex exponential in dest.
References complex_exp_gen_conj().
Referenced by complex_exp_gen_conj().

10.7.2.3.18 findUtmZone()

```
int findUtmZone (
            double latitude_deg,
            double longitude_deg)
```

Function finds the UTM zone number for given longitude and latitude.
References findUtmZone().
Referenced by findUtmZone().

10.7.2.3.19 front()

```
template<class T>
T & Gnss_circular_deque< T >::front (
            unsigned int ch)
```

Returns a reference to the first element in the deque.
Definition at line 81 of file gnss_circular_deque.h.

10.7.2.3.20 galileo_e1_code_gen_complex_sampled() [1/2]

```
void galileo_e1_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            const std::array< char, 3 > & signal_id,
            bool cboc,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

galileo_e1_code_gen_complex_sampled without secondary_flag for backward compatibility.
References galileo_e1_code_gen_complex_sampled().

10.7.2.3.21 galileo_e1_code_gen_complex_sampled() [2/2]

```
void galileo_e1_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            const std::array< char, 3 > & signal_id,
            bool cboc,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift,
            bool secondary_flag)
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).
References galileo_e1_code_gen_complex_sampled().
Referenced by galileo_e1_code_gen_complex_sampled(), and galileo_e1_code_gen_complex_sampled().

10.7.2.3.22 galileo_e1_code_gen_float_sampled() [1/2]

```
void galileo_e1_code_gen_float_sampled (
            own::span< float > dest,
            const std::array< char, 3 > & signal_id,
            bool cboc,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).
References galileo_e1_code_gen_float_sampled().

10.7.2.3.23 galileo_e1_code_gen_float_sampled() [2/2]

```
void galileo_e1_code_gen_float_sampled (
            own::span< float > dest,
            const std::array< char, 3 > & signal_id,
            bool cboc,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift,
            bool secondary_flag)
```

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling_freq).
References galileo_e1_code_gen_float_sampled().
Referenced by galileo_e1_code_gen_float_sampled(), and galileo_e1_code_gen_float_sampled().

10.7.2.3.24 galileo_e1_code_gen_sinboc11_float()

```
void galileo_e1_code_gen_sinboc11_float (
            own::span< float > dest,
            const std::array< char, 3 > & signal_id,
            uint32_t prn)
```

This function generates Galileo E1 code (can select E1B or E1C sinboc).
References galileo_e1_code_gen_sinboc11_float().
Referenced by galileo_e1_code_gen_sinboc11_float().

10.7.2.3.25 galileo_e5_a_code_gen_complex_primary()

```
void galileo_e5_a_code_gen_complex_primary (
            own::span< std::complex< float > > dest,
            int32_t prn,
            const std::array< char, 3 > & signal_id)
```

Generates Galileo E5a code at 1 sample/chip.
References galileo_e5_a_code_gen_complex_primary().
Referenced by galileo_e5_a_code_gen_complex_primary().

10.7.2.3.26 galileo_e5_a_code_gen_complex_sampled()

```
void galileo_e5_a_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            const std::array< char, 3 > & signal_id,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates Galileo E5a complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
References galileo_e5_a_code_gen_complex_sampled().
Referenced by galileo_e5_a_code_gen_complex_sampled().

10.7.2.3.27 galileo_e5_b_code_gen_complex_primary()

```
void galileo_e5_b_code_gen_complex_primary (
            own::span< std::complex< float > > dest,
            int32_t prn,
            const std::array< char, 3 > & signal_id)
```

Generates Galileo E5b code at 1 sample/chip.
References galileo_e5_b_code_gen_complex_primary().
Referenced by galileo_e5_b_code_gen_complex_primary().

10.7.2.3.28 galileo_e5_b_code_gen_complex_sampled()

```
void galileo_e5_b_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            const std::array< char, 3 > & signal_id,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates Galileo E5b complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
References galileo_e5_b_code_gen_complex_sampled().
Referenced by galileo_e5_b_code_gen_complex_sampled().

10.7.2.3.29 galileo_e6_b_code_gen_complex_primary()

```
void galileo_e6_b_code_gen_complex_primary (
            own::span< std::complex< float > > dest,
            int32_t prn)
```

Generates Galileo E6B code at 1 sample/chip.
References galileo_e6_b_code_gen_complex_primary().
Referenced by galileo_e6_b_code_gen_complex_primary().

10.7.2.3.30 galileo_e6_b_code_gen_complex_sampled()

```
void galileo_e6_b_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates Galileo E6B complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
References galileo_e6_b_code_gen_complex_sampled().
Referenced by galileo_e6_b_code_gen_complex_sampled().

10.7.2.3.31 galileo_e6_b_code_gen_float_primary()

```
void galileo_e6_b_code_gen_float_primary (
            own::span< float > dest,
            int32_t prn)
```

Generates Galileo E6B code at 1 sample/chip.
References galileo_e6_b_code_gen_float_primary().
Referenced by galileo_e6_b_code_gen_float_primary().

10.7.2.3.32 galileo_e6_c_code_gen_complex_primary()

```
void galileo_e6_c_code_gen_complex_primary (
            own::span< std::complex< float > > dest,
            int32_t prn)
```

Generates Galileo E6C codes at 1 sample/chip.
References galileo_e6_c_code_gen_complex_primary().
Referenced by galileo_e6_c_code_gen_complex_primary().

10.7.2.3.33 galileo_e6_c_code_gen_complex_sampled()

```
void galileo_e6_c_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates Galileo E6C complex codes, shifted to the desired chip and sampled at a frequency sampling_freq.
References galileo_e6_c_code_gen_complex_sampled().
Referenced by galileo_e6_c_code_gen_complex_sampled().

10.7.2.3.34 galileo_e6_c_code_gen_float_primary()

```
void galileo_e6_c_code_gen_float_primary (
            own::span< float > dest,
            int32_t prn)
```

Generates Galileo E6C codes at 1 sample/chip.
References galileo_e6_c_code_gen_float_primary().
Referenced by galileo_e6_c_code_gen_float_primary().

10.7.2.3.35 galileo_e6_c_secondary_code()

```
std::string galileo_e6_c_secondary_code (
            int32_t prn)
```

Generates a string with Galileo E6C secondary codes at 1 sample/chip.
References galileo_e6_c_secondary_code().
Referenced by galileo_e6_c_secondary_code().

10.7.2.3.36 galileo_e6_c_secondary_code_gen_complex()

```
void galileo_e6_c_secondary_code_gen_complex (
            own::span< std::complex< float > > dest,
            int32_t prn)
```

Generates Galileo E6C secondary codes at 1 sample/chip.
References galileo_e6_c_secondary_code_gen_complex().
Referenced by galileo_e6_c_secondary_code_gen_complex().

10.7.2.3.37 galileo_e6_c_secondary_code_gen_float()

```
void galileo_e6_c_secondary_code_gen_float (
            own::span< float > dest,
            int32_t prn)
```

Generates Galileo E6C secondary codes at 1 sample/chip.
References galileo_e6_c_secondary_code_gen_float().
Referenced by galileo_e6_c_secondary_code_gen_float().

10.7.2.3.38 Geo_to_ECEF()

```
void Geo_to_ECEF (
            const arma::vec & LLH,
            const arma::vec & v_eb_n,
            const arma::mat & C_b_n,
            arma::vec & r_eb_e,
            arma::vec & v_eb_e,
            arma::mat & C_b_e)
```

From Geographic to ECEF coordinates.
Inputs: LLH latitude (rad), longitude (rad), height (m) v_eb_n velocity of body frame w.r.t. ECEF frame, resolved along north, east, and down (m/s) C_b_n body-to-NED coordinate transformation matrix
Outputs: r_eb_e Cartesian position of body frame w.r.t. ECEF frame, resolved along ECEF-frame axes (m) v_eb↩_e velocity of body frame w.r.t. ECEF frame, resolved along ECEF-frame axes (m/s) C_b_e body-to-ECEF-frame coordinate transformation matrix
References Geo_to_ECEF().
Referenced by Geo_to_ECEF().

10.7.2.3.39 get()

```
template<class T>
const T & Gnss_circular_deque< T >::get (
            unsigned int ch,
            unsigned int pos) const
```

Returns a const reference to an element without bound checking.
Definition at line 95 of file gnss_circular_deque.h.

10.7.2.3.40 glonass_l1_ca_code_gen_complex()

```
void glonass_l1_ca_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t chip_shift)
```

Generates complex GLONASS L1 C/A code for the desired SV ID and code shift.

References glonass_l1_ca_code_gen_complex().
Referenced by glonass_l1_ca_code_gen_complex().

10.7.2.3.41 glonass_l1_ca_code_gen_complex_sampled()

```
void glonass_l1_ca_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates complex GLONASS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
References glonass_l1_ca_code_gen_complex_sampled().
Referenced by glonass_l1_ca_code_gen_complex_sampled().

10.7.2.3.42 glonass_l1_ca_code_gen_float()

```
void glonass_l1_ca_code_gen_float (
            own::span< float > dest,
            uint32_t chip_shift)
```

Generates float GLONASS L1 C/A code for the desired SV ID and code shift.
References glonass_l1_ca_code_gen_float().
Referenced by glonass_l1_ca_code_gen_float().

10.7.2.3.43 glonass_l2_ca_code_gen_complex()

```
void glonass_l2_ca_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t chip_shift)
```

Generates complex GLONASS L2 C/A code for the desired SV ID and code shift.
References glonass_l2_ca_code_gen_complex().
Referenced by glonass_l2_ca_code_gen_complex().

10.7.2.3.44 glonass_l2_ca_code_gen_complex_sampled()

```
void glonass_l2_ca_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates complex GLONASS L2 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
References glonass_l2_ca_code_gen_complex_sampled().
Referenced by glonass_l2_ca_code_gen_complex_sampled().

10.7.2.3.45 glonass_l2_ca_code_gen_float()

```
void glonass_l2_ca_code_gen_float (
            own::span< float > dest,
            uint32_t chip_shift)
```

Generates float GLONASS L2 C/A code for the desired SV ID and code shift.
References glonass_l2_ca_code_gen_float().
Referenced by glonass_l2_ca_code_gen_float().

10.7.2.3.46 Gnss_circular_deque() [1/2]

```
template<class T>
Gnss_circular_deque< T >::Gnss_circular_deque ()
```

Default constructor.
Definition at line 53 of file gnss_circular_deque.h.
References reset().

10.7.2.3.47 Gnss_circular_deque() [2/2]

```
template<class T>
Gnss_circular_deque< T >::Gnss_circular_deque (
            unsigned int max_size,
            unsigned int nchann)
```

nchann = number of channels; max_size = channel capacity
Definition at line 60 of file gnss_circular_deque.h.
References reset().

10.7.2.3.48 gps_l1_ca_code_gen_complex()

```
void gps_l1_ca_code_gen_complex (
            own::span< std::complex< float > > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates complex GPS L1 C/A code for the desired SV ID and code shift.
References gps_l1_ca_code_gen_complex().
Referenced by gps_l1_ca_code_gen_complex().

10.7.2.3.49 gps_l1_ca_code_gen_complex_sampled()

```
void gps_l1_ca_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq,
            uint32_t chip_shift)
```

Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.
References gps_l1_ca_code_gen_complex_sampled().
Referenced by gps_l1_ca_code_gen_complex_sampled().

10.7.2.3.50 gps_l1_ca_code_gen_float()

```
void gps_l1_ca_code_gen_float (
            own::span< float > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates float GPS L1 C/A code for the desired SV ID and code shift.
References gps_l1_ca_code_gen_float().
Referenced by gps_l1_ca_code_gen_float().

10.7.2.3.51 gps_l1_ca_code_gen_int()

```
void gps_l1_ca_code_gen_int (
            own::span< int32_t > dest,
            int32_t prn,
            uint32_t chip_shift)
```

Generates int GPS L1 C/A code for the desired SV ID and code shift.
References gps_l1_ca_code_gen_int().
Referenced by gps_l1_ca_code_gen_int().

10.7.2.3.52 gps_l2c_m_code_gen_complex()

```
void gps_l2c_m_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t prn)
```

Generates complex GPS L2C M code for the desired SV ID.
References gps_l2c_m_code_gen_complex().
Referenced by gps_l2c_m_code_gen_complex().

10.7.2.3.53 gps_l2c_m_code_gen_complex_sampled() [1/2]

```
void gps_l2c_m_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency.
References gps_l2c_m_code_gen_complex_sampled().
Referenced by gps_l2c_m_code_gen_complex_sampled(), and gps_l2c_m_code_gen_complex_sampled().

10.7.2.3.54 gps_l2c_m_code_gen_complex_sampled() [2/2]

```
void gps_l2c_m_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq,
            bool cl_zeroed)
```

Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency, optionally zeroing CL slots.
References gps_l2c_m_code_gen_complex_sampled().

10.7.2.3.55 gps_l2c_m_code_gen_float()

```
void gps_l2c_m_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates float GPS L2C M code for the desired SV ID.
References gps_l2c_m_code_gen_float().
Referenced by gps_l2c_m_code_gen_float().

10.7.2.3.56 gps_l2c_m_code_gen_float_cl_zeroed()

```
void gps_l2c_m_code_gen_float_cl_zeroed (
            own::span< float > dest,
            uint32_t prn)
```

Generates float GPS L2C M code with double chip rate and L component zeroed for the desired SV ID.
References gps_l2c_m_code_gen_float_cl_zeroed().
Referenced by gps_l2c_m_code_gen_float_cl_zeroed().

10.7.2.3.57 gps_l5i_code_gen_complex()

```
void gps_l5i_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t prn)
```

Generates complex GPS L5I code for the desired SV ID.
References gps_l5i_code_gen_complex().
Referenced by gps_l5i_code_gen_complex().

10.7.2.3.58 gps_l5i_code_gen_complex_sampled()

```
void gps_l5i_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex GPS L5I code for the desired SV ID, and sampled to specific sampling frequency.
References gps_l5i_code_gen_complex_sampled().
Referenced by gps_l5i_code_gen_complex_sampled().

10.7.2.3.59 gps_l5i_code_gen_float()

```
void gps_l5i_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real GPS L5I code for the desired SV ID.
References gps_l5i_code_gen_float().
Referenced by gps_l5i_code_gen_float().

10.7.2.3.60 gps_l5q_code_gen_complex()

```
void gps_l5q_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t prn)
```

Generates complex GPS L5Q code for the desired SV ID.
References gps_l5q_code_gen_complex().
Referenced by gps_l5q_code_gen_complex().

10.7.2.3.61 gps_l5q_code_gen_complex_sampled()

```
void gps_l5q_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex GPS L5Q code for the desired SV ID, and sampled to specific sampling frequency.
References gps_l5q_code_gen_complex_sampled().
Referenced by gps_l5q_code_gen_complex_sampled().

10.7.2.3.62 gps_l5q_code_gen_float()

```
void gps_l5q_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real GPS L5Q code for the desired SV ID.
References gps_l5q_code_gen_float().
Referenced by gps_l5q_code_gen_float().

10.7.2.3.63 Gravity_ECEF()

```
arma::vec Gravity_ECEF (
            const arma::vec & r_eb_e)
```

Calculates acceleration due to gravity resolved about ECEF-frame.
References Gravity_ECEF().
Referenced by Gravity_ECEF().

10.7.2.3.64 great_circle_distance()

```
double great_circle_distance (
            double lat1,
            double lon1,
            double lat2,
            double lon2)
```

The Haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes.
References great_circle_distance().
Referenced by great_circle_distance().

10.7.2.3.65 hex_to_binary_converter()

```
void hex_to_binary_converter (
            own::span< int32_t > dest,
            char from)
```

This function makes a conversion from hex (the input is a char) to binary (the output are 4 ints with +1 or -1 values).
References hex_to_binary_converter().
Referenced by hex_to_binary_converter().

10.7.2.3.66 hex_to_binary_string()

```
std::string hex_to_binary_string (
            char from)
```

This function makes a conversion from hex (the input is a char) to binary (the output is a string of 4 char with 0 or 1 values).
References hex_to_binary_string().
Referenced by hex_to_binary_string().

10.7.2.3.67 item_type_is_complex()

```
bool item_type_is_complex (
            const std::string & item_type)
```

Determine if an item_type is complex.
References item_type_is_complex().
Referenced by item_type_is_complex().

10.7.2.3.68 item_type_size()

```
size_t item_type_size (
            const std::string & item_type)
```

Return the size of the given item type, or zero if unknown.
References item_type_size().
Referenced by item_type_size().

10.7.2.3.69 item_type_valid()

```
bool item_type_valid (
            const std::string & item_type)
```

Check if a string is a valid item type.
\description Valid item types include: "byte", "short", "float", "ibyte", "ishort", "cbyte", "cshort", "gr_complex"
References item_type_valid().
Referenced by item_type_valid().

10.7.2.3.70 make_vector_converter()

```
item_type_converter_t make_vector_converter (
            const std::string & input_type,
            const std::string & output_type)
```

Create a function to convert an array of input_type to an array of output_type.
\description Provides a generic interface to generate conversion functions for mapping arrays of items.

Parameters

input_type	- String representation of the input item type
output_type	- String representation of the output item type

The item types accepted are:

1. "byte" for 8 bit integers

2. "cbyte" for complex (interleaved) 8 bit integers
3. "ibyte" for complex (interleaved) 8 bit integers
1. "short" for 16 bit integers
2. "cshort" for complex (interleaved) 16 bit integers
3. "ishort" for complex (interleaved) 16 bit integers
4. "float" for 32 bit floating point values
5. "gr_complex" for complex (interleaved) 32 bit floating point values

Returns

A function object with the following prototype: void convert_fun(void ∗dest, void ∗src, int num_items);

References make_vector_converter().
Referenced by make_vector_converter().

10.7.2.3.71 pop_front()

```
template<class T>
void Gnss_circular_deque< T >::pop_front (
             unsigned int ch)
```

Removes the first element of the deque.
Definition at line 130 of file gnss_circular_deque.h.

10.7.2.3.72 push_back()

```
template<class T>
void Gnss_circular_deque< T >::push_back (
             unsigned int ch,
             const T & new_data)
```

Inserts an element at the end of the deque.
Definition at line 137 of file gnss_circular_deque.h.

10.7.2.3.73 pv_Geo_to_ECEF()

```
void pv_Geo_to_ECEF (
             double L_b,
             double lambda_b,
             double h_b,
             const arma::vec & v_eb_n,
             arma::vec & r_eb_e,
             arma::vec & v_eb_e)
```

Converts curvilinear to Cartesian position and velocity resolving axes from NED to ECEF This function created 11/4/2012 by Paul Groves.
Inputs: L_b latitude (rad) lambda_b longitude (rad) h_b height (m) v_eb_n velocity of body frame w.r.t. ECEF frame, resolved along north, east, and down (m/s)
Outputs: r_eb_e Cartesian position of body frame w.r.t. ECEF frame, resolved along ECEF-frame axes (m) v_eb_e velocity of body frame w.r.t. ECEF frame, resolved along ECEF-frame axes (m/s)
References pv_Geo_to_ECEF().
Referenced by pv_Geo_to_ECEF().

10.7.2.3.74 qzss_l1_code_gen_complex_sampled()

```
void qzss_l1_code_gen_complex_sampled (
             own::span< std::complex< float > > dest,
             uint32_t prn,
             int32_t sampling_freq)
```

Generates complex QZSS L1 C/A(B) code for the desired SV ID, and sampled to specific sampling frequency.
References qzss_l1_code_gen_complex_sampled().
Referenced by qzss_l1_code_gen_complex_sampled().

10.7.2.3.75 qzss_l1_code_gen_float()

```
void qzss_l1_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real QZSS L1 C/A(B) code for the desired SV ID (QZSS_L1_SAMPLES_PER_CHIP samples per chip).
References qzss_l1_code_gen_float().
Referenced by qzss_l1_code_gen_float().

10.7.2.3.76 qzss_l1_code_is_boc()

```
bool qzss_l1_code_is_boc (
            uint32_t prn)
```

Returns true if the given QZSS PRN broadcasts the BOC(1,1)-modulated L1 C/B signal (IS-QZSS-PNT-006).
References qzss_l1_code_is_boc().
Referenced by qzss_l1_code_is_boc().

10.7.2.3.77 qzss_l5i_code_gen_complex_sampled()

```
void qzss_l5i_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex QZSS L5I code for the desired SV ID, and sampled to specific sampling frequency.
References qzss_l5i_code_gen_complex_sampled().
Referenced by qzss_l5i_code_gen_complex_sampled().

10.7.2.3.78 qzss_l5i_code_gen_float()

```
void qzss_l5i_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real QZSS L5I code for the desired SV ID.
References qzss_l5i_code_gen_float().
Referenced by qzss_l5i_code_gen_float().

10.7.2.3.79 qzss_l5q_code_gen_complex_sampled()

```
void qzss_l5q_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex QZSS L5Q code for the desired SV ID, and sampled to specific sampling frequency.
References qzss_l5q_code_gen_complex_sampled().
Referenced by qzss_l5q_code_gen_complex_sampled().

10.7.2.3.80 qzss_l5q_code_gen_float()

```
void qzss_l5q_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real QZSS L5I code for the desired SV ID.
References qzss_l5q_code_gen_float().
Referenced by qzss_l5q_code_gen_float().

10.7.2.3.81 resample_sensor_data_tags()

```
std::vector< gr::tag_t > resample_sensor_data_tags (
            const std::vector< gr::tag_t > & tags,
            double freq_in,
            double freq_out)
```

Updates timestamp within sensor data tags. To be used in resampler blocks.

Parameters

tags	Stream tags as retrieved by `get_tags_in_window` or `get_tags_in_range`.
freq_in	Input RF sample rate.
freq_out	Output RF sample rate.

Returns

Stream tags to be added with `add_item_tag`.

References resample_sensor_data_tags().
Referenced by resample_sensor_data_tags().

10.7.2.3.82 resampler() [1/2]

```
void resampler (
            const own::span< float > from,
            own::span< float > dest,
            float fs_in,
            float fs_out)
```

This function resamples a sequence of float values.
References resampler().
Referenced by resampler(), and resampler().

10.7.2.3.83 resampler() [2/2]

```
void resampler (
            own::span< const std::complex< float > > from,
            own::span< std::complex< float > > dest,
            float fs_in,
            float fs_out)
```

This function resamples a sequence of complex values.
References resampler().

10.7.2.3.84 reset() [1/2]

```
template<class T>
void Gnss_circular_deque< T >::reset ()
```

Removes all the channels (Sets nchann to 0).
Definition at line 123 of file gnss_circular_deque.h.

10.7.2.3.85 reset() [2/2]

```
template<class T>
void Gnss_circular_deque< T >::reset (
            unsigned int max_size,
            unsigned int nchann)
```

Removes all the elements in all the channels. Re-sets the number of channels and their capacity.
Definition at line 109 of file gnss_circular_deque.h.
Referenced by Gnss_circular_deque(), and Gnss_circular_deque().

10.7.2.3.86 sbas_l1_code_gen_complex()

```
void sbas_l1_code_gen_complex (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            uint32_t chip_shift)
```

Generates complex SBAS L1 C/A code for the desired SV PRN with a given chip_shift.
References sbas_l1_code_gen_complex().
Referenced by sbas_l1_code_gen_complex().

10.7.2.3.87 sbas_l1_code_gen_complex_sampled()

```
void sbas_l1_code_gen_complex_sampled (
            own::span< std::complex< float > > dest,
            uint32_t prn,
            int32_t sampling_freq)
```

Generates complex SBAS L1 C/A code for the desired SV PRN, sampled at sampling_freq [Hz].
References sbas_l1_code_gen_complex_sampled().
Referenced by sbas_l1_code_gen_complex_sampled().

10.7.2.3.88 sbas_l1_code_gen_float()

```
void sbas_l1_code_gen_float (
            own::span< float > dest,
            uint32_t prn)
```

Generates real SBAS L1 C/A code for the desired SV PRN (chip_shift=0).
References sbas_l1_code_gen_float().
Referenced by sbas_l1_code_gen_float().

10.7.2.3.89 size()

```
template<class T>
unsigned int Gnss_circular_deque< T >::size (
            unsigned int ch) const
```

Returns the number of available elements in a channel.
Definition at line 67 of file gnss_circular_deque.h.

10.7.2.3.90 Skew_symmetric()

```
arma::mat Skew_symmetric (
            const arma::vec & a)
```

Calculates skew-symmetric matrix.
References Skew_symmetric().
Referenced by Skew_symmetric().

10.7.2.3.91 togeod()

```
int togeod (
            double * dphi,
            double * dlambda,
            double * h,
            double a,
            double finv,
            double X,
            double Y,
            double Z)
```

Subroutine to calculate geodetic coordinates latitude, longitude, height given Cartesian coordinates X,Y,Z, and reference ellipsoid values semi-major axis (a) and the inverse of flattening (finv).
The output units of angular quantities will be in decimal degrees (15.5 degrees not 15 deg 30 min). The output units of h will be the same as the units of X,Y,Z,a.

```
  Inputs:
      a           - semi-major axis of the reference ellipsoid
      finv        - inverse of flattening of the reference ellipsoid
      X,Y,Z       - Cartesian coordinates

  Outputs:
      dphi        - latitude
      dlambda     - longitude
      h           - height above reference ellipsoid

      Based in a Matlab function by Kai Borre
```

References togeod().
Referenced by togeod().

10.7.2.3.92 topocent()

```
int topocent (
             double * Az,
             double * El,
             double * D,
             const arma::vec & x,
             const arma::vec & dx)
```

Transformation of vector dx into topocentric coordinate system with origin at x Inputs: x - vector origin coordinates (in ECEF system [X; Y; Z;]) dx - vector ([dX; dY; dZ;]).
Outputs: D - vector length. Units like the input Az - azimuth from north positive clockwise, degrees El - elevation angle, degrees
Based on a Matlab function by Kai Borre
References topocent().
Referenced by topocent().

10.7.3 gnss_sdr_flags

Functions

- ABSL_DECLARE_FLAG (std::string, c)

  Path to the configuration file.
- ABSL_DECLARE_FLAG (std::string, config_file)

  Path to the configuration file.
- ABSL_DECLARE_FLAG (std::string, log_dir)

  Path to the folder in which logging will be stored.
- ABSL_DECLARE_FLAG (std::string, s)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (std::string, signal_source)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (std::string, timestamp_source)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (bool, rf_shutdown)

  Shutdown RF when program exits.
- ABSL_DECLARE_FLAG (int32_t, doppler_max)

  If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, doppler_step)

  If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, cn0_samples)

  Number of correlator outputs used for CN0 estimation.
- ABSL_DECLARE_FLAG (int32_t, cn0_min)

  Minimum valid CN0 (in dB-Hz).
- ABSL_DECLARE_FLAG (int32_t, max_lock_fail)

Maximum number of code lock failures before dropping a satellite.

- ABSL_DECLARE_FLAG (int32_t, max_carrier_lock_fail)

  Maximum number of carrier lock failures before dropping a satellite.
- ABSL_DECLARE_FLAG (double, carrier_lock_th)

  Carrier lock threshold (in rad).
- ABSL_DECLARE_FLAG (double, dll_bw_hz)

  Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (double, pll_bw_hz)

  Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, carrier_smoothing_factor)

  Sets carrier smoothing factor M (overrides the configuration file).
- ABSL_DECLARE_FLAG (std::string, RINEX_version)

  If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.
- ABSL_DECLARE_FLAG (std::string, RINEX_name)

  If defined, specifies the RINEX files base name.
- ABSL_DECLARE_FLAG (bool, keyboard)

  If set to false, disables the keyboard listener. Only for debug purposes (e.g. ASAN mode termination).
- bool **ValidateFlags** ()

Variables

- const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200

10.7.3.1 Detailed Description

Library for command-line handling.

10.7.3.2 Function Documentation

10.7.3.2.1 ABSL_DECLARE_FLAG() [1/20]

```
ABSL_DECLARE_FLAG (
            bool ,
            keyboard )
```

If set to false, disables the keyboard listener. Only for debug purposes (e.g. ASAN mode termination).
References ABSL_DECLARE_FLAG().

10.7.3.2.2 ABSL_DECLARE_FLAG() [2/20]

```
ABSL_DECLARE_FLAG (
            bool ,
            rf_shutdown )
```

Shutdown RF when program exits.
References ABSL_DECLARE_FLAG().

10.7.3.2.3 ABSL_DECLARE_FLAG() [3/20]

```
ABSL_DECLARE_FLAG (
            double ,
            carrier_lock_th )
```

Carrier lock threshold (in rad).
References ABSL_DECLARE_FLAG().

10.7.3.2.4 ABSL_DECLARE_FLAG() [4/20]

```
ABSL_DECLARE_FLAG (
            double ,
            dll_bw_hz )
```

Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).
References ABSL_DECLARE_FLAG().

10.7.3.2.5 ABSL_DECLARE_FLAG() [5/20]

```
ABSL_DECLARE_FLAG (
            double ,
            pll_bw_hz )
```

Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).
References ABSL_DECLARE_FLAG().

10.7.3.2.6 ABSL_DECLARE_FLAG() [6/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            carrier_smoothing_factor )
```

Sets carrier smoothing factor M (overrides the configuration file).
References ABSL_DECLARE_FLAG().

10.7.3.2.7 ABSL_DECLARE_FLAG() [7/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            cn0_min )
```

Minimum valid CN0 (in dB-Hz).
References ABSL_DECLARE_FLAG().

10.7.3.2.8 ABSL_DECLARE_FLAG() [8/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            cn0_samples )
```

Number of correlator outputs used for CN0 estimation.
References ABSL_DECLARE_FLAG().

10.7.3.2.9 ABSL_DECLARE_FLAG() [9/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            doppler_max )
```

If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).
References ABSL_DECLARE_FLAG().

10.7.3.2.10 ABSL_DECLARE_FLAG() [10/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            doppler_step )
```

If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).
References ABSL_DECLARE_FLAG().

10.7.3.2.11 ABSL_DECLARE_FLAG() [11/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
```

```
            max_carrier_lock_fail )
```

Maximum number of carrier lock failures before dropping a satellite.
References ABSL_DECLARE_FLAG().

10.7.3.2.12 ABSL_DECLARE_FLAG() [12/20]

```
ABSL_DECLARE_FLAG (
            int32_t ,
            max_lock_fail )
```

Maximum number of code lock failures before dropping a satellite.
References ABSL_DECLARE_FLAG().

10.7.3.2.13 ABSL_DECLARE_FLAG() [13/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            c )
```

Path to the configuration file.
References ABSL_DECLARE_FLAG().
Referenced by ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), ABSL_DECLARE_FLAG(), and ABSL_DECLARE_FLAG().

10.7.3.2.14 ABSL_DECLARE_FLAG() [14/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            config_file )
```

Path to the configuration file.
References ABSL_DECLARE_FLAG().

10.7.3.2.15 ABSL_DECLARE_FLAG() [15/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            log_dir )
```

Path to the folder in which logging will be stored.
References ABSL_DECLARE_FLAG().

10.7.3.2.16 ABSL_DECLARE_FLAG() [16/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            RINEX_name )
```

If defined, specifies the RINEX files base name.
References ABSL_DECLARE_FLAG().

10.7.3.2.17 ABSL_DECLARE_FLAG() [17/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            RINEX_version )
```

If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.
References ABSL_DECLARE_FLAG().

10.7.3.2.18 ABSL_DECLARE_FLAG() [18/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            s )
```

Path to the file containing the signal samples.
References ABSL_DECLARE_FLAG().

10.7.3.2.19 ABSL_DECLARE_FLAG() [19/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            signal_source )
```

Path to the file containing the signal samples.
References ABSL_DECLARE_FLAG().

10.7.3.2.20 ABSL_DECLARE_FLAG() [20/20]

```
ABSL_DECLARE_FLAG (
            std::string ,
            timestamp_source )
```

Path to the file containing the signal samples.
References ABSL_DECLARE_FLAG().

10.7.3.3 Variable Documentation

10.7.3.3.1 DEFAULT_CARRIER_SMOOTHING_FACTOR

```
const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200
```

Definition at line 103 of file gnss_sdr_flags.h.

10.8 PVT

Topics

- algorithms_libs_rtklib
- RTKLIB
- pvt_adapters
- pvt_gr_blocks
- pvt_libs

10.8.1 Detailed Description

Computation of Position, Velocity and Time from GNSS observables.

10.8.2 algorithms_libs_rtklib

Classes

- struct gtime_t
- struct obsd_t
- struct obs_t
- struct erpd_t
- struct erp_t
- struct pcv_t
- struct pcvs_t
- struct alm_t
- struct eph_t
- struct geph_t

- struct peph_t
- struct pclk_t
- struct seph_t
- struct tled_t
- struct tle_t
- struct tec_t
- struct fcbd_t
- struct sbsmsg_t
- struct sbs_t
- struct sbsfcorr_t
- struct sbslcorr_t
- struct sbssatp_t
- struct sbssat_t
- struct sbsigp_t
- struct sbsigpband_t
- struct sbsion_t
- struct dgps_t
- struct ssr_t
- struct lexmsg_t
- struct lex_t
- struct lexeph_t
- struct lexion_t
- struct stec_t
- struct trop_t
- struct pppcorr_t
- struct nav_t
- struct sta_t
- struct sol_t
- struct solbuf_t
- struct solstat_t
- struct solstatbuf_t
- struct rtcm_t
- struct url_t
- struct opt_t
- struct exterr_t
- struct snrmask_t
- struct prcopt_t
- struct solopt_t
- struct ssat_t
- struct ambc_t
- struct rtk_t
- struct half_cyc_tag
- struct stream_t
- struct serial_t
- struct file_t
- struct tcp_t
- struct tcpsvr_t
- struct tcpcli_t
- struct ntrip_t
- struct ftp_t
- struct raw_t
- struct rtksvr_t
- struct msm_h_t
- class HAS_clock_corrections
- class HAS_orbit_corrections
- class HAS_bias_corrections
- class HAS_obs_corrections

Macros

- #define dev_t int
- #define socket_t int
- #define closesocket close
- #define lock_t pthread_mutex_t
- #define initlock(f)
- #define rtk_lock(f)
- #define rtk_unlock(f)
- #define VER_RTKLIB "2.4.2"
- #define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
- #define NTRIP_CLI_PORT 2101
- #define NTRIP_SVR_PORT 80
- #define NTRIP_MAXRSP 32768
- #define NTRIP_MAXSTR 256
- #define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"
- #define NTRIP_RSP_OK_SVR "OK\r\n"
- #define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n"
- #define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
- #define NTRIP_RSP_HTTP "HTTP/"
- #define NTRIP_RSP_ERROR "ERROR"
- #define FTP_CMD "wget"
- #define ENAGLO
- #define ENAQZS
- #define ENABDS
- #define STR_MODE_R 0x1
- #define STR_MODE_W 0x2
- #define STR_MODE_RW 0x3
- #define STR_NONE 0
- #define STR_SERIAL 1
- #define STR_FILE 2
- #define STR_TCPSVR 3
- #define STR_TCPCLI 4
- #define STR_UDP 5
- #define STR_NTRIPSVR 6
- #define STR_NTRIPCLI 7
- #define STR_FTP 8
- #define STR_HTTP 9
- #define NP_PPP(opt)
- #define IC_PPP(s, opt)
- #define IT_PPP(opt)
- #define NR_PPP(opt)
- #define IB_PPP(s, opt)
- #define NX_PPP(opt)
- #define NF_RTK(opt)
- #define NP_RTK(opt)
- #define NI_RTK(opt)
- #define NT_RTK(opt)
- #define NL_RTK(opt)
- #define NB_RTK(opt)
- #define NR_RTK(opt)
- #define NX_RTK(opt)
- #define II_RTK(s, opt)
- #define IT_RTK(r, opt)
- #define IL_RTK(f, opt)
- #define IB_RTK(s, f, opt)
- #define COMMENTH "%"
- #define MSG_DISCONN "$_DISCONNECT\r\n"

Typedefs

- using fatalfunc_t = void(const char *)

  fatal callback function type
- typedef struct half_cyc_tag **half_cyc_t**

Functions

- double sysefact (int sys)

  Measurement error factor of a navigation system (single home of the sys-to-EFACT mapping used by the single-point, relative and PPP error models).
- eph_t **eph_to_rtklib** (const Galileo_Ephemeris &gal_eph)
- eph_t **eph_to_rtklib** (const Galileo_Ephemeris &gal_eph, const std::map< int, HAS_orbit_corrections > &orbit_correction_map, const std::map< int, HAS_clock_corrections > &clock_correction_map)
- eph_t **eph_to_rtklib** (const Gps_Ephemeris &gps_eph, int ref_week=0)
- eph_t **eph_to_rtklib** (const Gps_Ephemeris &gps_eph, const std::map< int, HAS_orbit_corrections > &orbit_correction_map, const std::map< int, HAS_clock_corrections > &clock_correction_map, int ref_week=0)
- eph_t **eph_to_rtklib** (const Gps_CNAV_Ephemeris &gps_cnav_eph)
- eph_t **eph_to_rtklib** (const Beidou_Dnav_Ephemeris &bei_eph)
- eph_t **eph_to_rtklib** (const Beidou_Cnav1_Ephemeris &bei_eph)
- gtime_t glonass_almanac_epoch (const Glonass_Gnav_Almanac &almanac)

  Absolute GPST epoch of a GLONASS almanac (N_4/N_A calendar day plus t_lambda_n_A, which is given in MSK = UTC+3).
- alm_t alm_to_rtklib (const Gps_Almanac &gps_alm, int ref_week=0)

  Converts almanac data to RTKLIB's alm_t format.
- alm_t **alm_to_rtklib** (const Galileo_Almanac &gal_alm, int ref_week=0)
- alm_t **alm_to_rtklib** (const Beidou_Dnav_Almanac &bei_alm)
- bool glonass_gnav_is_healthy (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, bool glonass_strict_health=true)

  Broadcast health of a GLONASS satellite according to its GNAV ephemeris.
- geph_t eph_to_rtklib (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, const Glonass_Gnav_Utc_Model &gnav_clock_model, bool glonass_strict_health=true)

  Transforms a Glonass_Gnav_Ephemeris to its RTKLIB counterpart.
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, const std::map< std::string, std::map< int, HAS_obs_corrections > > &has_obs_corr, int week, int band, const HAS_obs_corrections **applied_has_correction, int ref_week)
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, const std::map< std::string, std::map< int, HAS_obs_corrections > > &has_obs_corr, int week, int band, int ref_week=0)
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, int week, int band, int ref_week=0)
- int **rtkopenstat** (const char *file, int level)
- void **rtkclosestat** ()
- void **rtkoutstat** (rtk_t *rtk)
- void **swapsolstat** ()
- void **outsolstat** (rtk_t *rtk)
- void **errmsg** (rtk_t *rtk, const char *format,...)
- double **sdobs** (const obsd_t *obs, int i, int j, int f)
- double **gfobs_L1L2** (const obsd_t *obs, int i, int j, const double *lam)
- double **gfobs_L1L5** (const obsd_t *obs, int i, int j, const double *lam)
- double **varerr** (int sat, int sys, double el, double snr_rover, double snr_base, double bl, double dt, int f, const prcopt_t *opt, const obsd_t *obs)
- double **baseline** (const double *ru, const double *rb, double *dr)
- void **initx_rtk** (rtk_t *rtk, double xi, double var, int i)
- int **selsat** (const obsd_t *obs, const double *azel, int nu, int nr, const prcopt_t *opt, int *sat, int *iu, int *ir)

- void **udpos** (rtk_t *rtk, double tt)
- void **udion** (rtk_t *rtk, double tt, double bl, const int *sat, int ns)
- void **udtrop** (rtk_t *rtk, double tt, double bl)
- void **udrcvbias** (rtk_t *rtk, double tt)
- void **detslp_ll** (rtk_t *rtk, const obsd_t *obs, int i, int rcv)
- void **detslp_gf_L1L2** (rtk_t *rtk, const obsd_t *obs, int i, int j, const nav_t *nav)
- void **detslp_gf_L1L5** (rtk_t *rtk, const obsd_t *obs, int i, int j, const nav_t *nav)
- void **detslp_dop** (rtk_t *rtk, const obsd_t *obs, const int *ix, int ns, int rcv, const nav_t *nav)
- void **udbias** (rtk_t *rtk, double tt, const obsd_t *obs, const int *sat, const int *iu, const int *ir, int ns, const nav_t *nav)
- void **udstate** (rtk_t *rtk, const obsd_t *obs, const int *sat, const int *iu, const int *ir, int ns, const nav_t *nav)
- void **zdres_sat** (int base, double r, const obsd_t *obs, const nav_t *nav, const double *azel, const double *dant, const prcopt_t *opt, double *y)
- int **zdres** (int base, const obsd_t *obs, int n, const double *rs, const double *dts, const int *svh, const nav_t *nav, const double *rr, const prcopt_t *opt, int index, double *y, double *e, double *azel)
- int **validobs** (int i, int j, int f, int nf, const double *y)
- void **ddcov** (const int *nb, int n, const double *Ri, const double *Rj, int nv, double *R)
- int **constbl** (rtk_t *rtk, const double *x, const double *P, double *v, double *H, double *Ri, double *Rj, int index)
- double **prectrop** (gtime_t time, const double *pos, int r, const double *azel, const prcopt_t *opt, const double *x, double *dtdx)
- double **gloicbcorr** (int sat1, int sat2, const prcopt_t *opt, double lam1, double lam2, int f)
- int **test_sys** (int sys, int m)
- int **ddres** (rtk_t *rtk, const obsd_t *obs, const nav_t *nav, double dt, const double *x, const double *P, const int *sat, double *y, const double *e, double *azel, const int *iu, const int *ir, int ns, double *v, double *H, double *R, int *vflg)
- double **intpres** (gtime_t time, const obsd_t *obs, int n, const nav_t *nav, rtk_t *rtk, double *y)
- int **ddidx** (rtk_t *rtk, int *ix, int glo, int sbs)
- void **restamb** (rtk_t *rtk, const double *bias, int nb, double *xa)
- void **holdamb** (rtk_t *rtk, const double *xa)
- int **resamb_LAMBDA** (rtk_t *rtk, double *bias, double *xa, int glo, int sbs)
- int **manage_amb_LAMBDA** (rtk_t *rtk, double *bias, double *xa, const int *sat, int nf, int ns)
- int **valpos** (rtk_t *rtk, const double *v, const double *R, const int *vflg, int nv, double thres)
- int **relpos** (rtk_t *rtk, const obsd_t *obs, int nu, int nr, const nav_t *nav)
- void **rtkinit** (rtk_t *rtk, const prcopt_t *opt)
- void **rtkfree** (rtk_t *rtk)
- int **rtkpos** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)
- const char * **opt2sep** (const solopt_t *opt)
- int **tonum** (char *buff, const char *sep, double *v)
- double **sqvar** (double covar)
- double **dmm2deg** (double dmm)
- void **septime** (double t, double *t1, double *t2, double *t3)
- void **soltocov** (const sol_t *sol, double *P)
- void **covtosol** (const double *P, sol_t *sol)
- int **decode_nmearmc** (char **val, int n, sol_t *sol)
- int **decode_nmeagga** (char **val, int n, sol_t *sol)
- int **decode_nmea** (char *buff, sol_t *sol)
- char * **decode_soltime** (char *buff, const solopt_t *opt, gtime_t *time)
- int **decode_solxyz** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solllh** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solenu** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solgsi** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solpos** (char *buff, const solopt_t *opt, sol_t *sol)
- void **decode_refpos** (char *buff, const solopt_t *opt, double *rb)
- int **decode_sol** (char *buff, const solopt_t *opt, sol_t *sol, double *rb)

- void **decode_solopt** (char *buff, solopt_t *opt)
- void **readsolopt** (FILE *fp, solopt_t *opt)
- int **inputsol** (unsigned char data, gtime_t ts, gtime_t te, double tint, int qflag, const solopt_t *opt, solbuf_t *solbuf)
- int **readsoldata** (FILE *fp, gtime_t ts, gtime_t te, double tint, int qflag, const solopt_t *opt, solbuf_t *solbuf)
- int **cmpsol** (const void *p1, const void *p2)
- int **sort_solbuf** (solbuf_t *solbuf)
- int **readsolt** (char *files[ ], int nfile, gtime_t ts, gtime_t te, double tint, int qflag, solbuf_t *solbuf)
- int **readsol** (char *files[ ], int nfile, solbuf_t *sol)
- int **addsol** (solbuf_t *solbuf, const sol_t *sol)
- sol_t * **getsol** (solbuf_t *solbuf, int index)
- void **initsolbuf** (solbuf_t *solbuf, int cyclic, int nmax)
- void **freesolbuf** (solbuf_t *solbuf)
- void **freesolstatbuf** (solstatbuf_t *solstatbuf)
- int **cmpsolstat** (const void *p1, const void *p2)
- int **sort_solstat** (solstatbuf_t *statbuf)
- int **decode_solstat** (char *buff, solstat_t *stat)
- void **addsolstat** (solstatbuf_t *statbuf, const solstat_t *stat)
- int **readsolstatdata** (FILE *fp, gtime_t ts, gtime_t te, double tint, solstatbuf_t *statbuf)
- int **readsolstatt** (char *files[ ], int nfile, gtime_t ts, gtime_t te, double tint, solstatbuf_t *statbuf)
- int **readsolstat** (char *files[ ], int nfile, solstatbuf_t *statbuf)
- int **outecef** (unsigned char *buff, const char *s, const sol_t *sol, const solopt_t *opt)
- int **outpos** (unsigned char *buff, const char *s, const sol_t *sol, const solopt_t *opt)
- int **outenu** (unsigned char *buff, const char *s, const sol_t *sol, const double *rb, const solopt_t *opt)
- int **outnmea_rmc** (unsigned char *buff, const sol_t *sol)
- int **outnmea_gga** (unsigned char *buff, const sol_t *sol)
- int **outnmea_gsa** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat)
- int **outnmea_gsv** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat)
- int **outprcopts** (unsigned char *buff, const prcopt_t *opt)
- int **outsolheads** (unsigned char *buff, const solopt_t *opt)
- int **outsols** (unsigned char *buff, const sol_t *sol, const double *rb, const solopt_t *opt)
- int **outsolexs** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat, const solopt_t *opt)
- void **outprcopt** (FILE *fp, const prcopt_t *opt)
- void **outsolhead** (FILE *fp, const solopt_t *opt)
- void **outsol** (FILE *fp, const sol_t *sol, const double *rb, const solopt_t *opt)
- void **outsolex** (FILE *fp, const sol_t *sol, const ssat_t *ssat, const solopt_t *opt)

Variables

- const int TINTACT = 200

  period for stream active (ms)
- const int SERIBUFFSIZE = 4096

  serial buffer size (bytes)
- const int TIMETAGH_LEN = 64

  time tag file header length
- const int MAXCLI = 32

  max client connection for tcp svr
- const int MAXSTATMSG = 32

  max length of status message
- const int FTP_TIMEOUT = 30

  ftp/http timeout (s)
- const int MAXRAWLEN = 4096

  max length of receiver raw message
- const int MAXSOLBUF = 256

max number of solution buffer

- const int MAXSBSMSG = 32

  max number of SBAS msg in RTK server

- const int MAXOBSBUF = 128

  max number of observation data buffer

- const int FILEPATHSEP = '/'
- const double RE_WGS84 = 6378137.0

  earth semimajor axis (WGS84) (m)

- const double FE_WGS84 = (1.0 / 298.257223563)

  earth flattening (WGS84)

- const double HION = 350000.0

  ionosphere height (m)

- const double PRN_HWBIAS = 1e-6

  process noise of h/w bias (m/MHz/sqrt(s))

- const double INT_SWAP_STAT = 86400.0

  swap interval of solution status file (s)

- const double INT_SWAP_TRAC = 86400.0

  swap interval of trace file (s)

- const unsigned int POLYCRC32 = 0xEDB88320u

  CRC32 polynomial.

- const unsigned int POLYCRC24Q = 0x1864CFBu

  CRC24Q polynomial.

- const int PMODE_SINGLE = 0

  positioning mode: single

- const int PMODE_DGPS = 1

  positioning mode: DGPS/DGNSS

- const int PMODE_KINEMA = 2

  positioning mode: kinematic

- const int PMODE_STATIC = 3

  positioning mode: static

- const int PMODE_MOVEB = 4

  positioning mode: moving-base

- const int PMODE_FIXED = 5

  positioning mode: fixed

- const int PMODE_PPP_KINEMA = 6

  positioning mode: PPP-kinemaric

- const int PMODE_PPP_STATIC = 7

  positioning mode: PPP-static

- const int PMODE_PPP_FIXED = 8

  positioning mode: PPP-fixed

- const int SOLF_LLH = 0

  solution format: lat/lon/height

- const int SOLF_XYZ = 1

  solution format: x/y/z-ecef

- const int SOLF_ENU = 2

  solution format: e/n/u-baseline

- const int SOLF_NMEA = 3

  solution format: NMEA-183

- const int SOLF_STAT = 4

  solution format: solution status

- const int SOLF_GSIF = 5

solution format: GSI F1/F2

- const int SOLQ_NONE = 0

  solution status: no solution
- const int SOLQ_FIX = 1

  solution status: fix
- const int SOLQ_FLOAT = 2

  solution status: float
- const int SOLQ_SBAS = 3

  solution status: SBAS
- const int SOLQ_DGPS = 4

  solution status: DGPS/DGNSS
- const int SOLQ_SINGLE = 5

  solution status: single
- const int SOLQ_PPP = 6

  solution status: PPP
- const int SOLQ_DR = 7

  solution status: dead reckoning
- const int MAXSOLQ = 7

  max number of solution status
- const int TIMES_GPST = 0

  time system: gps time
- const int TIMES_UTC = 1

  time system: utc
- const int TIMES_JST = 2

  time system: jst
- const double ERR_SAAS = 0.3

  saastamoinen model error std (m)
- const double ERR_BRDCI = 0.5

  broadcast iono model error factor
- const double ERR_CBIAS = 0.3

  code bias error std (m)
- const double REL_HUMI = 0.7

  relative humidity for saastamoinen model
- const double GAP_RESION = 120

  default gap to reset ionos parameters (ep)
- const int MAXFREQ = 7

  max NFREQ
- const int MAXLEAPS = 64

  max number of leap seconds table
- const double DTTOL = 0.005

  tolerance of time difference (s)
- const int NFREQ = 3

  number of carrier frequencies
- const int NFREQGLO = 2

  number of carrier frequencies of GLONASS
- const int NEXOBS = 0

  number of extended obs codes
- const int MAXANT = 64

  max length of station name/antenna type
- const int MINPRNGPS = 1

  min satellite PRN number of GPS

- const int MAXPRNGPS = 32

  max satellite PRN number of GPS
- const int NSATGPS = (MAXPRNGPS - MINPRNGPS + 1)

  number of GPS satellites
- const int NSYSGPS = 1
- const int SYS_NONE = 0x00

  navigation system: none
- const int SYS_GPS = 0x01

  navigation system: GPS
- const int SYS_SBS = 0x02

  navigation system: SBAS
- const int SYS_GLO = 0x04

  navigation system: GLONASS
- const int SYS_GAL = 0x08

  navigation system: Galileo
- const int SYS_QZS = 0x10

  navigation system: QZSS
- const int SYS_BDS = 0x20

  navigation system: BeiDou
- const int SYS_IRN = 0x40

  navigation system: IRNS
- const int SYS_LEO = 0x80

  navigation system: LEO
- const int SYS_ALL = 0xFF

  navigation system: all
- const int MINPRNGLO = 1

  min satellite slot number of GLONASS
- const int MAXPRNGLO = 27

  max satellite slot number of GLONASS
- const int NSATGLO = (MAXPRNGLO - MINPRNGLO + 1)

  number of GLONASS satellites
- const int NSYSGLO = 1
- const int MINPRNGAL = 1

  min satellite PRN number of Galileo
- const int MAXPRNGAL = 36

  max satellite PRN number of Galileo
- const int NSATGAL = (MAXPRNGAL - MINPRNGAL + 1)

  number of Galileo satellites
- const int NSYSGAL = 1
- const int MAXPRNQZS = 206

  max satellite PRN number of QZSS
- const int MINPRNQZS = 193

  min satellite PRN number of QZSS
- const int MINPRNQZS_S = 183

  min satellite PRN number of QZSS SAIF
- const int MAXPRNQZS_S = 191

  max satellite PRN number of QZSS SAIF
- const int NSATQZS = (MAXPRNQZS - MINPRNQZS + 1)

  number of QZSS satellites
- const int NSYSQZS = 1
- const int MINPRNBDS = 1

min satellite sat number of BeiDou

- const int MAXPRNBDS = 63

  max satellite sat number of BeiDou
- const int NSATBDS = (MAXPRNBDS - MINPRNBDS + 1)

  number of BeiDou satellites
- const int NSYSBDS = 1
- const int MINPRNIRN = 1

  min satellite sat number of IRNSS
- const int MAXPRNIRN = 7

  max satellite sat number of IRNSS
- const int NSATIRN = 0
- const int NSYSIRN = 0
- const int MINPRNLEO = 1

  min satellite sat number of LEO
- const int MAXPRNLEO = 10

  max satellite sat number of LEO ∗/
- const int NSATLEO = 0
- const int NSYSLEO = 0
- const int NSYS = (NSYSGPS + NSYSGLO + NSYSGAL + NSYSQZS + NSYSBDS + NSYSIRN + NSYSLEO)

  number of systems
- const int MINPRNSBS = 120

  min satellite PRN number of SBAS
- const int MAXPRNSBS = 142

  max satellite PRN number of SBAS
- const int NSATSBS = (MAXPRNSBS - MINPRNSBS + 1)

  number of SBAS satellites
- const int MAXSAT = (NSATGPS + NSATGLO + NSATGAL + NSATQZS + NSATBDS + NSATIRN + NSATSBS + NSATLEO)
- const int MAXSTA = 255
- const int MAXOBS = 64

  max number of obs in an epoch
- const int MAXRCV = 64

  max receiver number (1 to MAXRCV)
- const int MAXOBSTYPE = 64

  max number of obs type in RINEX
- const double MAXDTOE = 7200.0

  max time difference to GPS Toe (s)
- const double MAXDTOE_QZS = 7200.0

  max time difference to QZSS Toe (s)
- const double MAXDTOE_GAL = 10800.0

  max time difference to Galileo Toe (s)
- const double MAXDTOE_BDS = 21600.0

  max time difference to BeiDou Toe (s)
- const double MAXDTOE_GLO = 1800.0

  max time difference to GLONASS Toe (s)
- const double MAXDTOE_SBS = 360.0

  max time difference to SBAS Toe (s)
- const double MAXDTOE_S = 86400.0

  max time difference to ephem toe (s) for other
- const double MAXGDOP = 300.0

  max GDOP

- const int MAXSBSURA = 8
  max URA of SBAS satellite
- const int MAXBAND = 10
  max SBAS band of IGP
- const int MAXNIGP = 201
  max number of IGP in SBAS band
- const int MAXNGEO = 4
  max number of GEO satellites
- const int MAXSOLMSG = 8191
  max length of solution message
- const int MAXERRMSG = 4096
  max length of error/warning message
- const int IONOOPT_OFF = 0
  ionosphere option: correction off
- const int IONOOPT_BRDC = 1
  ionosphere option: broadcast model
- const int IONOOPT_SBAS = 2
  ionosphere option: SBAS model
- const int IONOOPT_IFLC = 3
  ionosphere option: L1/L2 or L1/L5 iono-free LC
- const int IONOOPT_EST = 4
  ionosphere option: estimation
- const int IONOOPT_TEC = 5
  ionosphere option: IONEX TEC model
- const int IONOOPT_QZS = 6
  ionosphere option: QZSS broadcast model
- const int IONOOPT_LEX = 7
  ionosphere option: QZSS LEX ionospehre
- const int IONOOPT_STEC = 8
  ionosphere option: SLANT TEC model
- const int TROPOPT_OFF = 0
  troposphere option: correction off
- const int TROPOPT_SAAS = 1
  troposphere option: Saastamoinen model
- const int TROPOPT_SBAS = 2
  troposphere option: SBAS model
- const int TROPOPT_EST = 3
  troposphere option: ZTD estimation
- const int TROPOPT_ESTG = 4
  troposphere option: ZTD+grad estimation
- const int TROPOPT_COR = 5
  troposphere option: ZTD correction
- const int TROPOPT_CORG = 6
  troposphere option: ZTD+grad correction
- const int EPHOPT_BRDC = 0
  ephemeris option: broadcast ephemeris
- const int EPHOPT_PREC = 1
  ephemeris option: precise ephemeris
- const int EPHOPT_SBAS = 2
  ephemeris option: broadcast + SBAS
- const int EPHOPT_SSRAPC = 3

ephemeris option: broadcast + SSR_APC

- const int EPHOPT_SSRCOM = 4

  ephemeris option: broadcast + SSR_COM
- const int EPHOPT_LEX = 5

  ephemeris option: QZSS LEX ephemeris
- constexpr double EFACT_GPS = 1.0

  error factor: GPS
- constexpr double EFACT_GLO = 1.5

  error factor: GLONASS
- constexpr double EFACT_GAL = 1.0

  error factor: Galileo
- constexpr double EFACT_QZS = 1.0

  error factor: QZSS
- constexpr double EFACT_BDS = 1.0

  error factor: BeiDou
- constexpr double EFACT_IRN = 1.5

  error factor: IRNSS
- constexpr double EFACT_SBS = 3.0

  error factor: SBAS
- const int MAXEXFILE = 1024

  max number of expanded files
- const double MAXSBSAGEF = 30.0

  max age of SBAS fast correction (s)
- const double MAXSBSAGEL = 1800.0

  max age of SBAS long term corr (s)
- const int ARMODE_OFF = 0

  AR mode: off.
- const int ARMODE_CONT = 1

  AR mode: continuous.
- const int ARMODE_INST = 2

  AR mode: instantaneous.
- const int ARMODE_FIXHOLD = 3

  AR mode: fix and hold.
- const int ARMODE_PPPAR = 4

  AR mode: PPP-AR.
- const int ARMODE_PPPAR_ILS = 5

  AR mode: AR mode: PPP-AR ILS.
- const int ARMODE_WLNL = 6
- const int ARMODE_TCAR = 7
- const int POSOPT_RINEX = 3

  pos option: rinex header pos
- const int MAXSTRPATH = 1024

  max length of stream path
- const int MAXSTRMSG = 1024

  max length of stream message
- const double CHISQR [100]
- const double LAM_CARR [MAXFREQ]
- const int STRFMT_RTCM2 = 0
- const int STRFMT_RTCM3 = 1
- const int STRFMT_SP3 = 16
- const int STRFMT_RNXCLK = 17

- const int STRFMT_SBAS = 18
- const int STRFMT_NMEA = 19
- const int MAXSTRRTK = 8
- const double VAR_POS = std::pow(30.0, 2.0)
- const double VAR_VEL = std::pow(10.0, 2.0)
- const double VAR_ACC = std::pow(10.0, 2.0)
- const double VAR_HWBIAS = std::pow(1.0, 2.0)
- const double VAR_GRA = std::pow(0.001, 2.0)
- const double INIT_ZWD = 0.15
- const double PRN_HWBIA = 1E-6
- const double MAXAC = 30.0
- const double VAR_HOLDAMB = 0.001
- const double TTOL_MOVEB = (1.0 + 2 ∗ DTTOL)

10.8.2.1 Detailed Description

Our version of the RTKLIB core library (see http://www.rtklib.com/)

10.8.2.2 Macro Definition Documentation

10.8.2.2.1 closesocket

```
#define closesocket close
```

Definition at line 59 of file rtklib.h.

10.8.2.2.2 COMMENTH

```
#define COMMENTH "%"
```

Definition at line 43 of file rtklib_solution.h.

10.8.2.2.3 dev_t

```
#define dev_t int
```

Definition at line 57 of file rtklib.h.

10.8.2.2.4 ENABDS

```
#define ENABDS
```

Definition at line 203 of file rtklib.h.

10.8.2.2.5 ENAGLO

```
#define ENAGLO
```

Definition at line 170 of file rtklib.h.

10.8.2.2.6 ENAQZS

```
#define ENAQZS
```

Definition at line 191 of file rtklib.h.

10.8.2.2.7 FTP_CMD

```
#define FTP_CMD "wget"
```

Definition at line 78 of file rtklib.h.

10.8.2.2.8 IB_PPP

```
#define IB_PPP(
               s,
               opt)
```

Value:

```
(NR_PPP(opt) + (s)-1)
```

Definition at line 369 of file rtklib.h.

10.8.2.2.9 IB_RTK

```
#define IB_RTK(
               s,
               f,
               opt)
```

Value:

```
(NR_RTK(opt) + MAXSAT * (f) + (s)-1)
```

Definition at line 68 of file rtklib_rtkpos.h.

10.8.2.2.10 IC_PPP

```
#define IC_PPP(
               s,
               opt)
```

Value:

```
(NP_PPP(opt) + (s))
```

Definition at line 366 of file rtklib.h.

10.8.2.2.11 II_RTK

```
#define II_RTK(
               s,
               opt)
```

Value:

```
(NP_RTK(opt) + (s)-1)
```

Definition at line 65 of file rtklib_rtkpos.h.

10.8.2.2.12 IL_RTK

```
#define IL_RTK(
               f,
               opt)
```

Value:

```
(NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + (f))
```

Definition at line 67 of file rtklib_rtkpos.h.

10.8.2.2.13 initlock

```
#define initlock(
               f)
```

Value:

```
pthread_mutex_init(f, NULL)
```

Definition at line 61 of file rtklib.h.

10.8.2.2.14 IT_PPP

```
#define IT_PPP(
               opt)
```

Value:

```
(IC_PPP(0, opt) + NSYS)
```

Definition at line 367 of file rtklib.h.

10.8.2.2.15 IT_RTK

```
#define IT_RTK(
              r,
              opt)
```

Value:

```
(NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) / 2 * (r))
```

Definition at line 66 of file rtklib_rtkpos.h.

10.8.2.2.16 lock_t

```
#define lock_t pthread_mutex_t
```

Definition at line 60 of file rtklib.h.

10.8.2.2.17 MSG_DISCONN

```
#define MSG_DISCONN "$_DISCONNECT\r\n"
```

Definition at line 44 of file rtklib_solution.h.

10.8.2.2.18 NB_RTK

```
#define NB_RTK(
              opt)
```

Value:

```
((opt)->mode <= PMODE_DGPS ? 0 : MAXSAT * NF_RTK(opt))
```

Definition at line 378 of file rtklib.h.

10.8.2.2.19 NF_RTK

```
#define NF_RTK(
              opt)
```

Value:

```
((opt)->ionoopt == IONOOPT_IFLC ? 1 : (opt)->nf)
```

Definition at line 373 of file rtklib.h.

10.8.2.2.20 NI_RTK

```
#define NI_RTK(
              opt)
```

Value:

```
((opt)->ionoopt != IONOOPT_EST ? 0 : MAXSAT)
```

Definition at line 375 of file rtklib.h.

10.8.2.2.21 NL_RTK

```
#define NL_RTK(
              opt)
```

Value:

```
((opt)->glomodear != 2 ? 0 : NFREQGLO)
```

Definition at line 377 of file rtklib.h.

10.8.2.2.22 NP_PPP

```
#define NP_PPP(
              opt)
```

Value:

```
((opt)->dynamics ? 9 : 3)
```

Definition at line 365 of file rtklib.h.

10.8.2.2.23 NP_RTK

```
#define NP_RTK(
               opt)
```

Value:

```
((opt)->dynamics == 0 ? 3 : 9)
```

Definition at line 374 of file rtklib.h.

10.8.2.2.24 NR_PPP

```
#define NR_PPP(
               opt)
```

Value:

```
(IT_PPP(opt) + ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt == TROPOPT_EST ? 1 : 3)))
```

Definition at line 368 of file rtklib.h.

10.8.2.2.25 NR_RTK

```
#define NR_RTK(
               opt)
```

Value:

```
(NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + NL_RTK(opt))
```

Definition at line 379 of file rtklib.h.

10.8.2.2.26 NT_RTK

```
#define NT_RTK(
               opt)
```

Value:

```
((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt < TROPOPT_ESTG ? 2 : 6))
```

Definition at line 376 of file rtklib.h.

10.8.2.2.27 NTRIP_AGENT

```
#define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
```

Definition at line 66 of file rtklib.h.

10.8.2.2.28 NTRIP_CLI_PORT

```
#define NTRIP_CLI_PORT 2101
```

Definition at line 67 of file rtklib.h.

10.8.2.2.29 NTRIP_MAXRSP

```
#define NTRIP_MAXRSP 32768
```

Definition at line 69 of file rtklib.h.

10.8.2.2.30 NTRIP_MAXSTR

```
#define NTRIP_MAXSTR 256
```

Definition at line 70 of file rtklib.h.

10.8.2.2.31 NTRIP_RSP_ERROR

```
#define NTRIP_RSP_ERROR "ERROR"
```

Definition at line 76 of file rtklib.h.

10.8.2.2.32 NTRIP_RSP_HTTP

```
#define NTRIP_RSP_HTTP "HTTP/"
```

Definition at line 75 of file rtklib.h.

10.8.2.2.33 NTRIP_RSP_OK_CLI

```
#define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"
```
Definition at line 71 of file rtklib.h.

10.8.2.2.34 NTRIP_RSP_OK_SVR

```
#define NTRIP_RSP_OK_SVR "OK\r\n"
```
Definition at line 72 of file rtklib.h.

10.8.2.2.35 NTRIP_RSP_SRCTBL

```
#define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n"
```
Definition at line 73 of file rtklib.h.

10.8.2.2.36 NTRIP_RSP_TBLEND

```
#define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
```
Definition at line 74 of file rtklib.h.

10.8.2.2.37 NTRIP_SVR_PORT

```
#define NTRIP_SVR_PORT 80
```
Definition at line 68 of file rtklib.h.

10.8.2.2.38 NX_PPP

```
#define NX_PPP(
              opt)
```
Value:
```
(IB_PPP(MAXSAT, opt) + 1)
```
Definition at line 370 of file rtklib.h.

10.8.2.2.39 NX_RTK

```
#define NX_RTK(
              opt)
```
Value:
```
(NR_RTK(opt) + NB_RTK(opt))
```
Definition at line 380 of file rtklib.h.

10.8.2.2.40 rtk_lock

```
#define rtk_lock(
              f)
```
Value:
```
pthread_mutex_lock(f)
```
Definition at line 62 of file rtklib.h.

10.8.2.2.41 rtk_unlock

```
#define rtk_unlock(
              f)
```
Value:
```
pthread_mutex_unlock(f)
```
Definition at line 63 of file rtklib.h.

10.8.2.2.42 socket_t

```
#define socket_t int
```
Definition at line 58 of file rtklib.h.

10.8.2.2.43 STR_FILE

```
#define STR_FILE 2
```
Definition at line 356 of file rtklib.h.

10.8.2.2.44 STR_FTP

```
#define STR_FTP 8
```
Definition at line 362 of file rtklib.h.

10.8.2.2.45 STR_HTTP

```
#define STR_HTTP 9
```
Definition at line 363 of file rtklib.h.

10.8.2.2.46 STR_MODE_R

```
#define STR_MODE_R 0x1
```
Definition at line 350 of file rtklib.h.

10.8.2.2.47 STR_MODE_RW

```
#define STR_MODE_RW 0x3
```
Definition at line 352 of file rtklib.h.

10.8.2.2.48 STR_MODE_W

```
#define STR_MODE_W 0x2
```
Definition at line 351 of file rtklib.h.

10.8.2.2.49 STR_NONE

```
#define STR_NONE 0
```
Definition at line 354 of file rtklib.h.

10.8.2.2.50 STR_NTRIPCLI

```
#define STR_NTRIPCLI 7
```
Definition at line 361 of file rtklib.h.

10.8.2.2.51 STR_NTRIPSVR

```
#define STR_NTRIPSVR 6
```
Definition at line 360 of file rtklib.h.

10.8.2.2.52 STR_SERIAL

```
#define STR_SERIAL 1
```
Definition at line 355 of file rtklib.h.

10.8.2.2.53 STR_TCPCLI

```
#define STR_TCPCLI 4
```
Definition at line 358 of file rtklib.h.

10.8.2.2.54 STR_TCPSVR

```
#define STR_TCPSVR 3
```
Definition at line 357 of file rtklib.h.

10.8.2.2.55 STR_UDP

```
#define STR_UDP 5
```
Definition at line 359 of file rtklib.h.

10.8.2.2.56 VER_RTKLIB

```
#define VER_RTKLIB "2.4.2"
```
Definition at line 65 of file rtklib.h.

10.8.2.3 Typedef Documentation

10.8.2.3.1 fatalfunc_t

```
using fatalfunc_t = void(const char *)
```
fatal callback function type
Definition at line 347 of file rtklib.h.

10.8.2.4 Function Documentation

10.8.2.4.1 alm_to_rtklib()

```
alm_t alm_to_rtklib (
            const Gps_Almanac & gps_alm,
            int ref_week = 0)
```
Converts almanac data to RTKLIB's alm_t format.

Parameters

ref_week	Current GPS week, used to resolve the truncated almanac week (WNa: 8 bits for GPS, 2 bits for Galileo) to the closest full week so that alm.toa is an absolute epoch, as eph_to_rtklib() does for ephemeris. 0 (default) uses WNa as-is; only valid for callers that never propagate the orbit (e.g. unit tests checking unrelated fields).

References alm_to_rtklib().
Referenced by alm_to_rtklib().

10.8.2.4.2 eph_to_rtklib()

```
geph_t eph_to_rtklib (
            const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            const Glonass_Gnav_Utc_Model & gnav_clock_model,
            bool glonass_strict_health = true)
```
Transforms a Glonass_Gnav_Ephemeris to its RTKLIB counterpart.

Parameters

glonass_gnav_eph	GLONASS GNAV Ephemeris structure
gnav_clock_model	GLONASS GNAV UTC model (for the GLONASST to GPST conversion)
glonass_strict_health	when true (default), the MSB of the Bn word also marks the satellite as unhealthy; when false, only the In flag is used

Returns

Ephemeris structure for RTKLIB parsing

10.8.2.4.3 glonass_almanac_epoch()

```
gtime_t glonass_almanac_epoch (
            const Glonass_Gnav_Almanac & almanac)
```

Absolute GPST epoch of a GLONASS almanac (N_4/N_A calendar day plus t_lambda_n_A, which is given in MSK = UTC+3).

Returns

Zero gtime_t when the broadcast date is unavailable or invalid.

References glonass_almanac_epoch().
Referenced by glonass_almanac_epoch().

10.8.2.4.4 glonass_gnav_is_healthy()

```
bool glonass_gnav_is_healthy (
            const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            bool glonass_strict_health = true)
```

Broadcast health of a GLONASS satellite according to its GNAV ephemeris.

Parameters

glonass_gnav_eph	GLONASS GNAV Ephemeris structure
glonass_strict_health	when true (default), the MSB of the Bn word also marks the satellite as unhealthy; when false, only the In flag is used

Returns

true if the satellite does not report a malfunction

References glonass_gnav_is_healthy().
Referenced by glonass_gnav_is_healthy().

10.8.2.4.5 sysefact()

```
double sysefact (
            int sys)  [inline]
```

Measurement error factor of a navigation system (single home of the sys-to-EFACT mapping used by the single-point, relative and PPP error models).
Definition at line 307 of file rtklib.h.
References EFACT_BDS, EFACT_GAL, EFACT_GLO, EFACT_GPS, EFACT_IRN, EFACT_QZS, EFACT_SBS, SYS_BDS, SYS_GAL, SYS_GLO, SYS_GPS, SYS_IRN, SYS_QZS, SYS_SBS, and sysefact().
Referenced by sysefact().

10.8.2.5 Variable Documentation

10.8.2.5.1 ARMODE_CONT

```
const int ARMODE_CONT = 1
```

AR mode: continuous.
Definition at line 334 of file rtklib.h.

10.8.2.5.2 ARMODE_FIXHOLD

```
const int ARMODE_FIXHOLD = 3
```

AR mode: fix and hold.
Definition at line 336 of file rtklib.h.

10.8.2.5.3 ARMODE_INST

```
const int ARMODE_INST = 2
```

AR mode: instantaneous.
Definition at line 335 of file rtklib.h.

10.8.2.5.4 ARMODE_OFF

```
const int ARMODE_OFF = 0
```

AR mode: off.
Definition at line 333 of file rtklib.h.

10.8.2.5.5 ARMODE_PPPAR

```
const int ARMODE_PPPAR = 4
```

AR mode: PPP-AR.
Definition at line 337 of file rtklib.h.

10.8.2.5.6 ARMODE_PPPAR_ILS

```
const int ARMODE_PPPAR_ILS = 5
```

AR mode: AR mode: PPP-AR ILS.
Definition at line 338 of file rtklib.h.

10.8.2.5.7 ARMODE_TCAR

```
const int ARMODE_TCAR = 7
```

Definition at line 340 of file rtklib.h.

10.8.2.5.8 ARMODE_WLNL

```
const int ARMODE_WLNL = 6
```

Definition at line 339 of file rtklib.h.

10.8.2.5.9 CHISQR

```
const double CHISQR[100]
```

Initial value:

```
                = {
10.8, 13.8, 16.3, 18.5, 20.5, 22.5, 24.3, 26.1, 27.9, 29.6,
31.3, 32.9, 34.5, 36.1, 37.7, 39.3, 40.8, 42.3, 43.8, 45.3,
46.8, 48.3, 49.7, 51.2, 52.6, 54.1, 55.5, 56.9, 58.3, 59.7,
61.1, 62.5, 63.9, 65.2, 66.6, 68.0, 69.3, 70.7, 72.1, 73.4,
74.7, 76.0, 77.3, 78.6, 80.0, 81.3, 82.6, 84.0, 85.4, 86.7,
88.0, 89.3, 90.6, 91.9, 93.3, 94.7, 96.0, 97.4, 98.7, 100,
101, 102, 103, 104, 105, 107, 108, 109, 110, 112,
113, 114, 115, 116, 118, 119, 120, 122, 123, 125,
126, 127, 128, 129, 131, 132, 133, 134, 135, 137,
138, 139, 140, 142, 143, 144, 145, 147, 148, 149}
```

Definition at line 1371 of file rtklib.h.

10.8.2.5.10 DTTOL

```
const double DTTOL = 0.005
```

tolerance of time difference (s)
Definition at line 146 of file rtklib.h.

10.8.2.5.11 EFACT_BDS

```
double EFACT_BDS = 1.0  [constexpr]
```

error factor: BeiDou
Definition at line 300 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.12 EFACT_GAL

```
double EFACT_GAL = 1.0  [constexpr]
```
error factor: Galileo
Definition at line 298 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.13 EFACT_GLO

```
double EFACT_GLO = 1.5  [constexpr]
```
error factor: GLONASS
Definition at line 297 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.14 EFACT_GPS

```
double EFACT_GPS = 1.0  [constexpr]
```
error factor: GPS
Definition at line 296 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.15 EFACT_IRN

```
double EFACT_IRN = 1.5  [constexpr]
```
error factor: IRNSS
Definition at line 301 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.16 EFACT_QZS

```
double EFACT_QZS = 1.0  [constexpr]
```
error factor: QZSS
Definition at line 299 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.17 EFACT_SBS

```
double EFACT_SBS = 3.0  [constexpr]
```
error factor: SBAS
Definition at line 302 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.18 EPHOPT_BRDC

```
const int EPHOPT_BRDC = 0
```
ephemeris option: broadcast ephemeris
Definition at line 288 of file rtklib.h.

10.8.2.5.19 EPHOPT_LEX

```
const int EPHOPT_LEX = 5
```
ephemeris option: QZSS LEX ephemeris
Definition at line 293 of file rtklib.h.

10.8.2.5.20 EPHOPT_PREC

```
const int EPHOPT_PREC = 1
```
ephemeris option: precise ephemeris
Definition at line 289 of file rtklib.h.

10.8.2.5.21 EPHOPT_SBAS

```
const int EPHOPT_SBAS = 2
```
ephemeris option: broadcast + SBAS
Definition at line 290 of file rtklib.h.

10.8.2.5.22 EPHOPT_SSRAPC

```
const int EPHOPT_SSRAPC = 3
```
ephemeris option: broadcast + SSR_APC
Definition at line 291 of file rtklib.h.

10.8.2.5.23 EPHOPT_SSRCOM

```
const int EPHOPT_SSRCOM = 4
```
ephemeris option: broadcast + SSR_COM
Definition at line 292 of file rtklib.h.

10.8.2.5.24 ERR_BRDCI

```
const double ERR_BRDCI = 0.5
```
broadcast iono model error factor
Definition at line 138 of file rtklib.h.

10.8.2.5.25 ERR_CBIAS

```
const double ERR_CBIAS = 0.3
```
code bias error std (m)
Definition at line 139 of file rtklib.h.

10.8.2.5.26 ERR_SAAS

```
const double ERR_SAAS = 0.3
```
saastamoinen model error std (m)
Definition at line 137 of file rtklib.h.

10.8.2.5.27 FE_WGS84

```
const double FE_WGS84 = (1.0 / 298.257223563)
```
earth flattening (WGS84)
Definition at line 94 of file rtklib.h.

10.8.2.5.28 FILEPATHSEP

```
const int FILEPATHSEP = '/'
```
Definition at line 92 of file rtklib.h.

10.8.2.5.29 FTP_TIMEOUT

```
const int FTP_TIMEOUT = 30
```
ftp/http timeout (s)
Definition at line 86 of file rtklib.h.

10.8.2.5.30 GAP_RESION

```
const double GAP_RESION = 120
```
default gap to reset ionos parameters (ep)
Definition at line 141 of file rtklib.h.

10.8.2.5.31 HION

```
const double HION = 350000.0
```
ionosphere height (m)
Definition at line 96 of file rtklib.h.

10.8.2.5.32 INIT_ZWD

```
const double INIT_ZWD = 0.15
```
Definition at line 50 of file rtklib_rtkpos.h.

10.8.2.5.33 INT_SWAP_STAT

```
const double INT_SWAP_STAT = 86400.0
```
swap interval of solution status file (s)
Definition at line 99 of file rtklib.h.

10.8.2.5.34 INT_SWAP_TRAC

```
const double INT_SWAP_TRAC = 86400.0
```
swap interval of trace file (s)
Definition at line 100 of file rtklib.h.

10.8.2.5.35 IONOOPT_BRDC

```
const int IONOOPT_BRDC = 1
```
ionosphere option: broadcast model
Definition at line 270 of file rtklib.h.

10.8.2.5.36 IONOOPT_EST

```
const int IONOOPT_EST = 4
```
ionosphere option: estimation
Definition at line 273 of file rtklib.h.

10.8.2.5.37 IONOOPT_IFLC

```
const int IONOOPT_IFLC = 3
```
ionosphere option: L1/L2 or L1/L5 iono-free LC
Definition at line 272 of file rtklib.h.

10.8.2.5.38 IONOOPT_LEX

```
const int IONOOPT_LEX = 7
```
ionosphere option: QZSS LEX ionospehre
Definition at line 276 of file rtklib.h.

10.8.2.5.39 IONOOPT_OFF

```
const int IONOOPT_OFF = 0
```
ionosphere option: correction off
Definition at line 269 of file rtklib.h.

10.8.2.5.40 IONOOPT_QZS

```
const int IONOOPT_QZS = 6
```
ionosphere option: QZSS broadcast model
Definition at line 275 of file rtklib.h.

10.8.2.5.41 IONOOPT_SBAS

```
const int IONOOPT_SBAS = 2
```
ionosphere option: SBAS model
Definition at line 271 of file rtklib.h.

10.8.2.5.42 IONOOPT_STEC

```
const int IONOOPT_STEC = 8
```
ionosphere option: SLANT TEC model
Definition at line 277 of file rtklib.h.

10.8.2.5.43 IONOOPT_TEC

```
const int IONOOPT_TEC = 5
```
ionosphere option: IONEX TEC model
Definition at line 274 of file rtklib.h.

10.8.2.5.44 LAM_CARR

```
const double LAM_CARR[MAXFREQ]
```
Initial value:
```
                  = {
SPEED_OF_LIGHT_M_S / FREQ1, SPEED_OF_LIGHT_M_S / FREQ2, SPEED_OF_LIGHT_M_S / FREQ5, SPEED_OF_LIGHT_M_S /
      FREQ6, SPEED_OF_LIGHT_M_S / FREQ7,
SPEED_OF_LIGHT_M_S / FREQ8, SPEED_OF_LIGHT_M_S / FREQ9}
```
Definition at line 1384 of file rtklib.h.

10.8.2.5.45 MAXAC

```
const double MAXAC = 30.0
```
Definition at line 53 of file rtklib_rtkpos.h.

10.8.2.5.46 MAXANT

```
const int MAXANT = 64
```
max length of station name/antenna type
Definition at line 151 of file rtklib.h.

10.8.2.5.47 MAXBAND

```
const int MAXBAND = 10
```
max SBAS band of IGP
Definition at line 262 of file rtklib.h.

10.8.2.5.48 MAXCLI

```
const int MAXCLI = 32
```
max client connection for tcp svr
Definition at line 83 of file rtklib.h.

10.8.2.5.49 MAXDTOE

```
const double MAXDTOE = 7200.0
```
max time difference to GPS Toe (s)
Definition at line 252 of file rtklib.h.

10.8.2.5.50 MAXDTOE_BDS

```
const double MAXDTOE_BDS = 21600.0
```
max time difference to BeiDou Toe (s)
Definition at line 255 of file rtklib.h.

10.8.2.5.51 MAXDTOE_GAL

```
const double MAXDTOE_GAL = 10800.0
```
max time difference to Galileo Toe (s)
Definition at line 254 of file rtklib.h.

10.8.2.5.52 MAXDTOE_GLO

```
const double MAXDTOE_GLO = 1800.0
```
max time difference to GLONASS Toe (s)
Definition at line 256 of file rtklib.h.

10.8.2.5.53 MAXDTOE_QZS

```
const double MAXDTOE_QZS = 7200.0
```
max time difference to QZSS Toe (s)
Definition at line 253 of file rtklib.h.

10.8.2.5.54 MAXDTOE_S

```
const double MAXDTOE_S = 86400.0
```
max time difference to ephem toe (s) for other
Definition at line 258 of file rtklib.h.

10.8.2.5.55 MAXDTOE_SBS

```
const double MAXDTOE_SBS = 360.0
```
max time difference to SBAS Toe (s)
Definition at line 257 of file rtklib.h.

10.8.2.5.56 MAXERRMSG

```
const int MAXERRMSG = 4096
```
max length of error/warning message
Definition at line 267 of file rtklib.h.

10.8.2.5.57 MAXEXFILE

```
const int MAXEXFILE = 1024
```
max number of expanded files
Definition at line 329 of file rtklib.h.

10.8.2.5.58 MAXFREQ

```
const int MAXFREQ = 7
```
max NFREQ
Definition at line 143 of file rtklib.h.

10.8.2.5.59 MAXGDOP

```
const double MAXGDOP = 300.0
```
max GDOP
Definition at line 259 of file rtklib.h.

10.8.2.5.60 MAXLEAPS

```
const int MAXLEAPS = 64
```
max number of leap seconds table
Definition at line 145 of file rtklib.h.

10.8.2.5.61 MAXNGEO

```
const int MAXNGEO = 4
```
max number of GEO satellites
Definition at line 264 of file rtklib.h.

10.8.2.5.62 MAXNIGP

```
const int MAXNIGP = 201
```
max number of IGP in SBAS band
Definition at line 263 of file rtklib.h.

10.8.2.5.63 MAXOBS

```
const int MAXOBS = 64
```
max number of obs in an epoch
Definition at line 247 of file rtklib.h.

10.8.2.5.64 MAXOBSBUF

```
const int MAXOBSBUF = 128
```
max number of observation data buffer
Definition at line 90 of file rtklib.h.

10.8.2.5.65 MAXOBSTYPE

```
const int MAXOBSTYPE = 64
```
max number of obs type in RINEX
Definition at line 251 of file rtklib.h.

10.8.2.5.66 MAXPRNBDS

```
const int MAXPRNBDS = 63
```
max satellite sat number of BeiDou
Definition at line 206 of file rtklib.h.

10.8.2.5.67 MAXPRNGAL

```
const int MAXPRNGAL = 36
```
max satellite PRN number of Galileo
Definition at line 185 of file rtklib.h.

10.8.2.5.68 MAXPRNGLO

```
const int MAXPRNGLO = 27
```
max satellite slot number of GLONASS
Definition at line 173 of file rtklib.h.

10.8.2.5.69 MAXPRNGPS

```
const int MAXPRNGPS = 32
```
max satellite PRN number of GPS
Definition at line 154 of file rtklib.h.

10.8.2.5.70 MAXPRNIRN

```
const int MAXPRNIRN = 7
```
max satellite sat number of IRNSS
Definition at line 217 of file rtklib.h.

10.8.2.5.71 MAXPRNLEO

```
const int MAXPRNLEO = 10
```
max satellite sat number of LEO */
Definition at line 227 of file rtklib.h.

10.8.2.5.72 MAXPRNQZS

```
const int MAXPRNQZS = 206
```
max satellite PRN number of QZSS
Definition at line 189 of file rtklib.h.

10.8.2.5.73 MAXPRNQZS_S

```
const int MAXPRNQZS_S = 191
```
max satellite PRN number of QZSS SAIF
Definition at line 194 of file rtklib.h.

10.8.2.5.74 MAXPRNSBS

```
const int MAXPRNSBS = 142
```
max satellite PRN number of SBAS
Definition at line 239 of file rtklib.h.

10.8.2.5.75 MAXRAWLEN

```
const int MAXRAWLEN = 4096
```
max length of receiver raw message
Definition at line 87 of file rtklib.h.

10.8.2.5.76 MAXRCV

```
const int MAXRCV = 64
```
max receiver number (1 to MAXRCV)
Definition at line 250 of file rtklib.h.

10.8.2.5.77 MAXSAT

```
const int MAXSAT = (NSATGPS + NSATGLO + NSATGAL + NSATQZS + NSATBDS + NSATIRN + NSATSBS +
NSATLEO)
```
Definition at line 242 of file rtklib.h.

10.8.2.5.78 MAXSBSAGEF

```
const double MAXSBSAGEF = 30.0
```
max age of SBAS fast correction (s)
Definition at line 330 of file rtklib.h.

10.8.2.5.79 MAXSBSAGEL

```
const double MAXSBSAGEL = 1800.0
```
max age of SBAS long term corr (s)
Definition at line 331 of file rtklib.h.

10.8.2.5.80 MAXSBSMSG

```
const int MAXSBSMSG = 32
```
max number of SBAS msg in RTK server
Definition at line 89 of file rtklib.h.

10.8.2.5.81 MAXSBSURA

```
const int MAXSBSURA = 8
```

max URA of SBAS satellite
Definition at line 261 of file rtklib.h.

10.8.2.5.82 MAXSOLBUF

```
const int MAXSOLBUF = 256
```

max number of solution buffer
Definition at line 88 of file rtklib.h.

10.8.2.5.83 MAXSOLMSG

```
const int MAXSOLMSG = 8191
```

max length of solution message
Definition at line 266 of file rtklib.h.

10.8.2.5.84 MAXSOLQ

```
const int MAXSOLQ = 7
```

max number of solution status
Definition at line 130 of file rtklib.h.

10.8.2.5.85 MAXSTA

```
const int MAXSTA = 255
```

Definition at line 244 of file rtklib.h.

10.8.2.5.86 MAXSTATMSG

```
const int MAXSTATMSG = 32
```

max length of status message
Definition at line 84 of file rtklib.h.

10.8.2.5.87 MAXSTRMSG

```
const int MAXSTRMSG = 1024
```

max length of stream message
Definition at line 345 of file rtklib.h.

10.8.2.5.88 MAXSTRPATH

```
const int MAXSTRPATH = 1024
```

max length of stream path
Definition at line 344 of file rtklib.h.

10.8.2.5.89 MAXSTRRTK

```
const int MAXSTRRTK = 8
```

Definition at line 1396 of file rtklib.h.

10.8.2.5.90 MINPRNBDS

```
const int MINPRNBDS = 1
```

min satellite sat number of BeiDou
Definition at line 205 of file rtklib.h.

10.8.2.5.91 MINPRNGAL

```
const int MINPRNGAL = 1
```

min satellite PRN number of Galileo
Definition at line 184 of file rtklib.h.

10.8.2.5.92 MINPRNGLO

```
const int MINPRNGLO = 1
```

min satellite slot number of GLONASS
Definition at line 172 of file rtklib.h.

10.8.2.5.93 MINPRNGPS

```
const int MINPRNGPS = 1
```

min satellite PRN number of GPS
Definition at line 153 of file rtklib.h.

10.8.2.5.94 MINPRNIRN

```
const int MINPRNIRN = 1
```

min satellite sat number of IRNSS
Definition at line 216 of file rtklib.h.

10.8.2.5.95 MINPRNLEO

```
const int MINPRNLEO = 1
```

min satellite sat number of LEO
Definition at line 226 of file rtklib.h.

10.8.2.5.96 MINPRNQZS

```
const int MINPRNQZS = 193
```

min satellite PRN number of QZSS
Definition at line 190 of file rtklib.h.

10.8.2.5.97 MINPRNQZS_S

```
const int MINPRNQZS_S = 183
```

min satellite PRN number of QZSS SAIF
Definition at line 193 of file rtklib.h.

10.8.2.5.98 MINPRNSBS

```
const int MINPRNSBS = 120
```

min satellite PRN number of SBAS
Definition at line 238 of file rtklib.h.

10.8.2.5.99 NEXOBS

```
const int NEXOBS = 0
```

number of extended obs codes
Definition at line 150 of file rtklib.h.

10.8.2.5.100 NFREQ

```
const int NFREQ = 3
```

number of carrier frequencies
Definition at line 148 of file rtklib.h.

10.8.2.5.101 NFREQGLO

```
const int NFREQGLO = 2
```

number of carrier frequencies of GLONASS
Definition at line 149 of file rtklib.h.

10.8.2.5.102 NSATBDS

```
const int NSATBDS = (MAXPRNBDS - MINPRNBDS + 1)
```

number of BeiDou satellites
Definition at line 207 of file rtklib.h.

10.8.2.5.103 NSATGAL

```
const int NSATGAL = (MAXPRNGAL - MINPRNGAL + 1)
```

number of Galileo satellites
Definition at line 186 of file rtklib.h.

10.8.2.5.104 NSATGLO

```
const int NSATGLO = (MAXPRNGLO - MINPRNGLO + 1)
```

number of GLONASS satellites
Definition at line 174 of file rtklib.h.

10.8.2.5.105 NSATGPS

```
const int NSATGPS = (MAXPRNGPS - MINPRNGPS + 1)
```

number of GPS satellites
Definition at line 155 of file rtklib.h.

10.8.2.5.106 NSATIRN

```
const int NSATIRN = 0
```

Definition at line 222 of file rtklib.h.

10.8.2.5.107 NSATLEO

```
const int NSATLEO = 0
```

Definition at line 232 of file rtklib.h.

10.8.2.5.108 NSATQZS

```
const int NSATQZS = (MAXPRNQZS - MINPRNQZS + 1)
```

number of QZSS satellites
Definition at line 195 of file rtklib.h.

10.8.2.5.109 NSATSBS

```
const int NSATSBS = (MAXPRNSBS - MINPRNSBS + 1)
```

number of SBAS satellites
Definition at line 240 of file rtklib.h.

10.8.2.5.110 NSYS

```
const int NSYS = (NSYSGPS + NSYSGLO + NSYSGAL + NSYSQZS + NSYSBDS + NSYSIRN + NSYSLEO)
```

number of systems
Definition at line 236 of file rtklib.h.

10.8.2.5.111 NSYSBDS

```
const int NSYSBDS = 1
```
Definition at line 208 of file rtklib.h.

10.8.2.5.112 NSYSGAL

```
const int NSYSGAL = 1
```
Definition at line 187 of file rtklib.h.

10.8.2.5.113 NSYSGLO

```
const int NSYSGLO = 1
```
Definition at line 175 of file rtklib.h.

10.8.2.5.114 NSYSGPS

```
const int NSYSGPS = 1
```
Definition at line 156 of file rtklib.h.

10.8.2.5.115 NSYSIRN

```
const int NSYSIRN = 0
```
Definition at line 223 of file rtklib.h.

10.8.2.5.116 NSYSLEO

```
const int NSYSLEO = 0
```
Definition at line 233 of file rtklib.h.

10.8.2.5.117 NSYSQZS

```
const int NSYSQZS = 1
```
Definition at line 196 of file rtklib.h.

10.8.2.5.118 PMODE_DGPS

```
const int PMODE_DGPS = 1
```
positioning mode: DGPS/DGNSS
Definition at line 106 of file rtklib.h.

10.8.2.5.119 PMODE_FIXED

```
const int PMODE_FIXED = 5
```
positioning mode: fixed
Definition at line 110 of file rtklib.h.

10.8.2.5.120 PMODE_KINEMA

```
const int PMODE_KINEMA = 2
```
positioning mode: kinematic
Definition at line 107 of file rtklib.h.

10.8.2.5.121 PMODE_MOVEB

```
const int PMODE_MOVEB = 4
```
positioning mode: moving-base
Definition at line 109 of file rtklib.h.

10.8.2.5.122 PMODE_PPP_FIXED

```
const int PMODE_PPP_FIXED = 8
```
positioning mode: PPP-fixed
Definition at line 113 of file rtklib.h.

10.8.2.5.123 PMODE_PPP_KINEMA

```
const int PMODE_PPP_KINEMA = 6
```
positioning mode: PPP-kinemaric
Definition at line 111 of file rtklib.h.

10.8.2.5.124 PMODE_PPP_STATIC

```
const int PMODE_PPP_STATIC = 7
```
positioning mode: PPP-static
Definition at line 112 of file rtklib.h.

10.8.2.5.125 PMODE_SINGLE

```
const int PMODE_SINGLE = 0
```
positioning mode: single
Definition at line 105 of file rtklib.h.

10.8.2.5.126 PMODE_STATIC

```
const int PMODE_STATIC = 3
```
positioning mode: static
Definition at line 108 of file rtklib.h.

10.8.2.5.127 POLYCRC24Q

```
const unsigned int POLYCRC24Q = 0x1864CFBu
```
CRC24Q polynomial.
Definition at line 103 of file rtklib.h.

10.8.2.5.128 POLYCRC32

```
const unsigned int POLYCRC32 = 0xEDB88320u
```
CRC32 polynomial.
Definition at line 102 of file rtklib.h.

10.8.2.5.129 POSOPT_RINEX

```
const int POSOPT_RINEX = 3
```
pos option: rinex header pos
Definition at line 343 of file rtklib.h.

10.8.2.5.130 PRN_HWBIA

```
const double PRN_HWBIA = 1E-6
```
Definition at line 52 of file rtklib_rtkpos.h.

10.8.2.5.131 PRN_HWBIAS

```
const double PRN_HWBIAS = 1e-6
```
process noise of h/w bias (m/MHz/sqrt(s))
Definition at line 97 of file rtklib.h.

10.8.2.5.132 RE_WGS84

```
const double RE_WGS84 = 6378137.0
```
earth semimajor axis (WGS84) (m)
Definition at line 93 of file rtklib.h.

10.8.2.5.133 REL_HUMI

```
const double REL_HUMI = 0.7
```
relative humidity for saastamoinen model
Definition at line 140 of file rtklib.h.

10.8.2.5.134 SERIBUFFSIZE

```
const int SERIBUFFSIZE = 4096
```
serial buffer size (bytes)
Definition at line 81 of file rtklib.h.

10.8.2.5.135 SOLF_ENU

```
const int SOLF_ENU = 2
```
solution format: e/n/u-baseline
Definition at line 117 of file rtklib.h.

10.8.2.5.136 SOLF_GSIF

```
const int SOLF_GSIF = 5
```
solution format: GSI F1/F2
Definition at line 120 of file rtklib.h.

10.8.2.5.137 SOLF_LLH

```
const int SOLF_LLH = 0
```
solution format: lat/lon/height
Definition at line 115 of file rtklib.h.

10.8.2.5.138 SOLF_NMEA

```
const int SOLF_NMEA = 3
```
solution format: NMEA-183
Definition at line 118 of file rtklib.h.

10.8.2.5.139 SOLF_STAT

```
const int SOLF_STAT = 4
```
solution format: solution status
Definition at line 119 of file rtklib.h.

10.8.2.5.140 SOLF_XYZ

```
const int SOLF_XYZ = 1
```
solution format: x/y/z-ecef
Definition at line 116 of file rtklib.h.

10.8.2.5.141 SOLQ_DGPS

```
const int SOLQ_DGPS = 4
```
solution status: DGPS/DGNSS
Definition at line 126 of file rtklib.h.

10.8.2.5.142 SOLQ_DR

```
const int SOLQ_DR = 7
```
solution status: dead reckoning
Definition at line 129 of file rtklib.h.

10.8.2.5.143 SOLQ_FIX

```
const int SOLQ_FIX = 1
```
solution status: fix
Definition at line 123 of file rtklib.h.

10.8.2.5.144 SOLQ_FLOAT

```
const int SOLQ_FLOAT = 2
```
solution status: float
Definition at line 124 of file rtklib.h.

10.8.2.5.145 SOLQ_NONE

```
const int SOLQ_NONE = 0
```
solution status: no solution
Definition at line 122 of file rtklib.h.

10.8.2.5.146 SOLQ_PPP

```
const int SOLQ_PPP = 6
```
solution status: PPP
Definition at line 128 of file rtklib.h.

10.8.2.5.147 SOLQ_SBAS

```
const int SOLQ_SBAS = 3
```
solution status: SBAS
Definition at line 125 of file rtklib.h.

10.8.2.5.148 SOLQ_SINGLE

```
const int SOLQ_SINGLE = 5
```
solution status: single
Definition at line 127 of file rtklib.h.

10.8.2.5.149 STRFMT_NMEA

```
const int STRFMT_NMEA = 19
```
Definition at line 1393 of file rtklib.h.

10.8.2.5.150 STRFMT_RNXCLK

```
const int STRFMT_RNXCLK = 17
```
Definition at line 1391 of file rtklib.h.

10.8.2.5.151 STRFMT_RTCM2

```
const int STRFMT_RTCM2 = 0
```
Definition at line 1388 of file rtklib.h.

10.8.2.5.152 STRFMT_RTCM3

```
const int STRFMT_RTCM3 = 1
```
Definition at line 1389 of file rtklib.h.

10.8.2.5.153 STRFMT_SBAS

```
const int STRFMT_SBAS = 18
```
Definition at line 1392 of file rtklib.h.

10.8.2.5.154 STRFMT_SP3

```
const int STRFMT_SP3 = 16
```
Definition at line 1390 of file rtklib.h.

10.8.2.5.155 SYS_ALL

```
const int SYS_ALL = 0xFF
```
navigation system: all
Definition at line 167 of file rtklib.h.

10.8.2.5.156 SYS_BDS

```
const int SYS_BDS = 0x20
```
navigation system: BeiDou
Definition at line 164 of file rtklib.h.
Referenced by ntrip_rtk_signals(), ntrip_rtk_system_label(), and sysefact().

10.8.2.5.157 SYS_GAL

```
const int SYS_GAL = 0x08
```
navigation system: Galileo
Definition at line 162 of file rtklib.h.
Referenced by ntrip_rtk_signals(), ntrip_rtk_system_label(), and sysefact().

10.8.2.5.158 SYS_GLO

```
const int SYS_GLO = 0x04
```
navigation system: GLONASS
Definition at line 161 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.159 SYS_GPS

```
const int SYS_GPS = 0x01
```
navigation system: GPS
Definition at line 159 of file rtklib.h.
Referenced by ntrip_rtk_second_band_slot(), ntrip_rtk_signals(), ntrip_rtk_system_label(), and sysefact().

10.8.2.5.160 SYS_IRN

```
const int SYS_IRN = 0x40
```
navigation system: IRNS
Definition at line 165 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.161 SYS_LEO

```
const int SYS_LEO = 0x80
```
navigation system: LEO
Definition at line 166 of file rtklib.h.

10.8.2.5.162 SYS_NONE

```
const int SYS_NONE = 0x00
```
navigation system: none
Definition at line 158 of file rtklib.h.

10.8.2.5.163 SYS_QZS

```
const int SYS_QZS = 0x10
```
navigation system: QZSS
Definition at line 163 of file rtklib.h.
Referenced by ntrip_rtk_second_band_slot(), ntrip_rtk_system_label(), and sysefact().

10.8.2.5.164 SYS_SBS

```
const int SYS_SBS = 0x02
```
navigation system: SBAS
Definition at line 160 of file rtklib.h.
Referenced by sysefact().

10.8.2.5.165 TIMES_GPST

```
const int TIMES_GPST = 0
```
time system: gps time
Definition at line 132 of file rtklib.h.

10.8.2.5.166 TIMES_JST

```
const int TIMES_JST = 2
```
time system: jst
Definition at line 134 of file rtklib.h.

10.8.2.5.167 TIMES_UTC

```
const int TIMES_UTC = 1
```
time system: utc
Definition at line 133 of file rtklib.h.

10.8.2.5.168 TIMETAGH_LEN

```
const int TIMETAGH_LEN = 64
```
time tag file header length
Definition at line 82 of file rtklib.h.

10.8.2.5.169 TINTACT

```
const int TINTACT = 200
```
period for stream active (ms)
Definition at line 80 of file rtklib.h.

10.8.2.5.170 TROPOPT_COR

```
const int TROPOPT_COR = 5
```
troposphere option: ZTD correction
Definition at line 284 of file rtklib.h.

10.8.2.5.171 TROPOPT_CORG

```
const int TROPOPT_CORG = 6
```
troposphere option: ZTD+grad correction
Definition at line 285 of file rtklib.h.

10.8.2.5.172 TROPOPT_EST

```
const int TROPOPT_EST = 3
```
troposphere option: ZTD estimation
Definition at line 282 of file rtklib.h.

10.8.2.5.173 TROPOPT_ESTG

```
const int TROPOPT_ESTG = 4
```
troposphere option: ZTD+grad estimation
Definition at line 283 of file rtklib.h.

10.8.2.5.174 TROPOPT_OFF

```
const int TROPOPT_OFF = 0
```
troposphere option: correction off
Definition at line 279 of file rtklib.h.

10.8.2.5.175 TROPOPT_SAAS

```
const int TROPOPT_SAAS = 1
```
troposphere option: Saastamoinen model
Definition at line 280 of file rtklib.h.

10.8.2.5.176 TROPOPT_SBAS

```
const int TROPOPT_SBAS = 2
```
troposphere option: SBAS model
Definition at line 281 of file rtklib.h.

10.8.2.5.177 TTOL_MOVEB

```
const double TTOL_MOVEB = (1.0 + 2 * DTTOL)
```
Definition at line 57 of file rtklib_rtkpos.h.

10.8.2.5.178 VAR_ACC

```
const double VAR_ACC = std::pow(10.0, 2.0)
```
Definition at line 47 of file rtklib_rtkpos.h.

10.8.2.5.179 VAR_GRA

```
const double VAR_GRA = std::pow(0.001, 2.0)
```
Definition at line 49 of file rtklib_rtkpos.h.

10.8.2.5.180 VAR_HOLDAMB

```
const double VAR_HOLDAMB = 0.001
```
Definition at line 55 of file rtklib_rtkpos.h.

10.8.2.5.181 VAR_HWBIAS

```
const double VAR_HWBIAS = std::pow(1.0, 2.0)
```
Definition at line 48 of file rtklib_rtkpos.h.

10.8.2.5.182 VAR_POS

```
const double VAR_POS = std::pow(30.0, 2.0)
```
Definition at line 45 of file rtklib_rtkpos.h.

10.8.2.5.183 VAR_VEL

```
const double VAR_VEL = std::pow(10.0, 2.0)
```
Definition at line 46 of file rtklib_rtkpos.h.

10.8.3 RTKLIB

Functions

- void **secure_wipe** (void *data, std::size_t size)

10.8.3.1 Detailed Description

10.8.4 pvt_adapters

Classes

- class Rtklib_Pvt

  This class implements a PvtInterface for the RTKLIB PVT block.

10.8.4.1 Detailed Description

Wrap GNU Radio PVT solvers with a PvtInterface

10.8.5 pvt_gr_blocks

Classes

- class rtklib_pvt_gs

  This class implements a block that computes the PVT solution using the RTKLIB integrated library.

Typedefs

- using rtklib_pvt_gs_sptr = gnss_shared_ptr<rtklib_pvt_gs>

Functions

- rtklib_pvt_gs_sptr **rtklib_make_pvt_gs** (uint32_t nchannels, const Pvt_Conf &conf_, const rtk_t &rtk, const SensorDataSourceConfiguration &sensor_data_configuration)

10.8.5.1 Detailed Description

GNU Radio blocks for the computation of PVT solutions.

10.8.5.2 Typedef Documentation

10.8.5.2.1 rtklib_pvt_gs_sptr

```
using rtklib_pvt_gs_sptr = gnss_shared_ptr<rtklib_pvt_gs>
```
Definition at line 85 of file rtklib_pvt_gs.h.

10.8.6 pvt_libs

Classes

- struct sdr_gnss_packet_t
- struct sdr_gnss_packet_t::[struct].sats
- class Serdes_Monitor_Pvt

  This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers.
- class Serdes_Gps_Eph

  This class implements serialization and deserialization of Gps_Ephemeris objects using Protocol Buffers.

- class Serdes_Galileo_Eph

  This class implements serialization and deserialization of Galileo_Ephemeris using Protocol Buffers.
- class Secure_String

  Value type for credential-bearing configuration fields.
- class Sbas_Rtklib_Corrections

  Maintains RTKLIB SBAS state from timestamped raw L1 messages.
- class Rtklib_Solver

  This class implements a PVT solution based on RTKLIB.
- class Rtcm_Printer

  This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.
- class Rtcm

  This class implements the generation and reading of some Message Types defined in the RTCM 3.2 Standard, plus some utilities to handle messages.
- class Rinex_Printer

  Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.
- class Pvt_Solution

  Base class for a PVT solution.
- class Pvt_Kf

  Kalman Filter for Position and Velocity.
- class Pvt_Conf
- struct Ntrip_Rtk_Signal
- class Ntrip_Rtcm_Client

  Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster, preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced v1, optionally over TLS.
- struct Ntrip_Rtcm_Snapshot
- struct Ntrip_Rtcm_Client_Status
- struct Ntrip_Rtcm_Client_Config
- class Nmea_Printer

  This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA).
- class Monitor_Pvt_Udp_Sink
- class Monitor_Pvt

  This class contains parameters and outputs of the PVT block.
- class Monitor_Ephemeris_Udp_Sink
- class Kml_Printer

  Prints PVT information to OGC KML format file (can be viewed with Google Earth).
- class Has_Simple_Printer

  Prints HAS messages content in a txt file. See HAS-SIS-ICD for a message description.
- class Gpx_Printer

  Prints PVT information to GPX format file.
- class GeoJSON_Printer

  Prints PVT solutions in GeoJSON format file.
- class Geohash

  Class for geohash encoding / decoding See https://en.wikipedia.org/wiki/Geohash.
- class An_Packet_Printer

  class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.
- struct an_packet_t

Typedefs

- using b_io_context = boost::asio::io_service
- using b_io_context = boost::asio::io_service
- using b_io_context = boost::asio::io_service

Enumerations

- enum class **Ntrip_Rtcm_Client_State** {
  DISABLED , **STOPPED** , **STARTING** , **CONNECTING** ,
  AUTHENTICATING , **STREAMING** , **RECONNECT_WAIT** , **ERROR** ,
  STOPPING }
- enum class **Rtklib_Fixed_Base_Status** {
  NOT_REQUESTED , **MISSING_OBSERVATIONS** , **MISSING_POSITION** , **STALE** ,
  INVALID_POSITION , **INSUFFICIENT_COMMON_SATELLITES** , **APPLIED** , **SOLVER_FALLBACK** ,
  SOLVER_FAILURE }

Functions

- void secure_clear_ntrip_credentials (Pvt_Conf ∗conf) noexcept

  Scrubs the NTRIP credential fields of a Pvt_Conf copy ahead of its destruction (the Secure_String fields wipe themselves on release, but a long-lived copy should drop its secrets as soon as they are consumed). Single home of the credential field list.
- bool ntrip_text_has_space_or_control (const std::string &text)

  True when the text contains whitespace or control characters — the character rule shared by the credential validation and the adapter's environment-variable-name check.
- Ntrip_Rtcm_Client_Config make_ntrip_rtcm_client_config (const Pvt_Conf &conf)

  Builds the NTRIP client configuration from the PVT configuration — the single home of the field mapping, shared by the adapter's validation and by the PVT block's client construction. The mountpoint is stored normalized; validate() and the transport-path builder normalize again defensively for directly-built configurations.
- bool ntrip_rtk_is_entry_band (const Ntrip_Rtk_Signal &signal)

  The entry band of every constellation occupies RTKLIB's first slot, so "first band" is band_slot == 0 by definition — a fact, not a column that could drift from the slot assignment.
- const std::array< Ntrip_Rtk_Signal, 6 > & ntrip_rtk_signals ()
- uint32_t ntrip_rtk_supported_signal_mask ()

  Bitmask of every signal_flag the NTRIP RTK path supports.
- int ntrip_rtk_navigation_systems (const Signal_Enabled_Flags &enabled)

  RTKLIB navigation-system mask spanned by the enabled NTRIP RTK channels.
- int ntrip_rtk_required_bands (const Signal_Enabled_Flags &enabled)

  Number of RTKLIB frequency slots the enabled NTRIP RTK channels require.
- int ntrip_rtk_second_band_slot (int system, const Signal_Enabled_Flags &enabled)
- const char ∗ ntrip_rtk_system_label (int system)

  Constellation label of an RTKLIB system bit, for messages.
- bool ntrip_rtk_valid_channel_set (const Signal_Enabled_Flags &enabled)
- std::string ntrip_rtk_supported_sets_description ()

  Human-readable list of the supported per-system channel sets, derived from the table for the adapter's rejection message.

10.8.6.1 Detailed Description

Library for the computation of PVT solutions.

10.8.6.2 Typedef Documentation

10.8.6.2.1 b_io_context [1/3]

```
using b_io_context = boost::asio::io_service
```

Definition at line 39 of file monitor_ephemeris_udp_sink.h.

10.8.6.2.2 b_io_context [2/3]

```
using b_io_context = boost::asio::io_service
```

Definition at line 37 of file monitor_pvt_udp_sink.h.

10.8.6.2.3 b_io_context [3/3]

```
using b_io_context = boost::asio::io_service
```
Definition at line 64 of file rtcm.h.

10.8.6.3 Enumeration Type Documentation

10.8.6.3.1 Ntrip_Rtcm_Client_State

```
enum class Ntrip_Rtcm_Client_State  [strong]
```
Definition at line 95 of file ntrip_rtcm_client.h.

10.8.6.3.2 Rtklib_Fixed_Base_Status

```
enum class Rtklib_Fixed_Base_Status  [strong]
```
Definition at line 95 of file rtklib_solver.h.

10.8.6.4 Function Documentation

10.8.6.4.1 make_ntrip_rtcm_client_config()

```
Ntrip_Rtcm_Client_Config make_ntrip_rtcm_client_config (
            const Pvt_Conf & conf)
```
Builds the NTRIP client configuration from the PVT configuration — the single home of the field mapping, shared by the adapter's validation and by the PVT block's client construction. The mountpoint is stored normalized; validate() and the transport-path builder normalize again defensively for directly-built configurations.
References make_ntrip_rtcm_client_config().
Referenced by make_ntrip_rtcm_client_config().

10.8.6.4.2 ntrip_rtk_is_entry_band()

```
bool ntrip_rtk_is_entry_band (
            const Ntrip_Rtk_Signal & signal)  [inline]
```
The entry band of every constellation occupies RTKLIB's first slot, so "first band" is band_slot == 0 by definition — a fact, not a column that could drift from the slot assignment.
Definition at line 58 of file ntrip_rtk_signals.h.
References Ntrip_Rtk_Signal::band_slot, and ntrip_rtk_is_entry_band().
Referenced by ntrip_rtk_is_entry_band(), ntrip_rtk_navigation_systems(), ntrip_rtk_second_band_slot(), ntrip_rtk_supported_sets_description(), and ntrip_rtk_valid_channel_set().

10.8.6.4.3 ntrip_rtk_navigation_systems()

```
int ntrip_rtk_navigation_systems (
            const Signal_Enabled_Flags & enabled)  [inline]
```
RTKLIB navigation-system mask spanned by the enabled NTRIP RTK channels.
Definition at line 98 of file ntrip_rtk_signals.h.
References ntrip_rtk_is_entry_band(), ntrip_rtk_navigation_systems(), and ntrip_rtk_signals().
Referenced by ntrip_rtk_navigation_systems().

10.8.6.4.4 ntrip_rtk_required_bands()

```
int ntrip_rtk_required_bands (
            const Signal_Enabled_Flags & enabled)  [inline]
```
Number of RTKLIB frequency slots the enabled NTRIP RTK channels require.
Definition at line 113 of file ntrip_rtk_signals.h.
References ntrip_rtk_required_bands(), and ntrip_rtk_signals().
Referenced by ntrip_rtk_required_bands().

10.8.6.4.5 ntrip_rtk_second_band_slot()

```
int ntrip_rtk_second_band_slot (
            int system,
            const Signal_Enabled_Flags & enabled)  [inline]
```

RTKLIB slot of a satellite system's second band given the enabled channels. QZSS rides the GPS bands. When no second band of the system is enabled, the constellation's canonical second slot is returned (Galileo/BeiDou: 2, both anticipate slot-2 signals; GPS/QZSS: 1) so unsupported slots can be cleared.
Definition at line 133 of file ntrip_rtk_signals.h.
References ntrip_rtk_is_entry_band(), ntrip_rtk_second_band_slot(), ntrip_rtk_signals(), SYS_GPS, and SYS_QZS.
Referenced by ntrip_rtk_second_band_slot().

10.8.6.4.6 ntrip_rtk_signals()

```
const std::array< Ntrip_Rtk_Signal, 6 > & ntrip_rtk_signals ()  [inline]
```

The signal table. A function-local static keeps the header buildable as C++11 (an inline variable would need C++17) while still providing a single instance. Keep each constellation's second bands ordered by preference: with more than one enabled (a configuration the adapter rejects today), ntrip_rtk_second_band_slot() picks the first match.
Definition at line 71 of file ntrip_rtk_signals.h.
References ntrip_rtk_signals(), SYS_BDS, SYS_GAL, and SYS_GPS.
Referenced by ntrip_rtk_navigation_systems(), ntrip_rtk_required_bands(), ntrip_rtk_second_band_slot(), ntrip_rtk_signals(), ntrip_rtk_supported_sets_description(), ntrip_rtk_supported_signal_mask(), and ntrip_rtk_valid_channel_set().

10.8.6.4.7 ntrip_rtk_supported_sets_description()

```
std::string ntrip_rtk_supported_sets_description ()  [inline]
```

Human-readable list of the supported per-system channel sets, derived from the table for the adapter's rejection message.
Definition at line 230 of file ntrip_rtk_signals.h.
References ntrip_rtk_is_entry_band(), ntrip_rtk_signals(), ntrip_rtk_supported_sets_description(), and ntrip_rtk_system_label().
Referenced by ntrip_rtk_supported_sets_description().

10.8.6.4.8 ntrip_rtk_supported_signal_mask()

```
uint32_t ntrip_rtk_supported_signal_mask ()  [inline]
```

Bitmask of every signal_flag the NTRIP RTK path supports.
Definition at line 86 of file ntrip_rtk_signals.h.
References ntrip_rtk_signals(), and ntrip_rtk_supported_signal_mask().
Referenced by ntrip_rtk_supported_signal_mask(), and ntrip_rtk_valid_channel_set().

10.8.6.4.9 ntrip_rtk_system_label()

```
const char * ntrip_rtk_system_label (
            int system)  [inline]
```

Constellation label of an RTKLIB system bit, for messages.
Definition at line 159 of file ntrip_rtk_signals.h.
References ntrip_rtk_system_label(), SYS_BDS, SYS_GAL, SYS_GPS, and SYS_QZS.
Referenced by ntrip_rtk_supported_sets_description(), and ntrip_rtk_system_label().

10.8.6.4.10 ntrip_rtk_valid_channel_set()

```
bool ntrip_rtk_valid_channel_set (
            const Signal_Enabled_Flags & enabled)  [inline]
```

True when the enabled channel set follows the NTRIP RTK rules, all of them derived from the table's columns: only signals the table lists, at least one constellation entry band, at most one second band per system, and every second band accompanied by its system's entry band.
Definition at line 183 of file ntrip_rtk_signals.h.
References ntrip_rtk_is_entry_band(), ntrip_rtk_signals(), ntrip_rtk_supported_signal_mask(), and ntrip_rtk_valid_channel_set().
Referenced by ntrip_rtk_valid_channel_set().

10.8.6.4.11 ntrip_text_has_space_or_control()

```
bool ntrip_text_has_space_or_control (
            const std::string & text)
```

True when the text contains whitespace or control characters — the character rule shared by the credential validation and the adapter's environment-variable-name check.
References ntrip_text_has_space_or_control().
Referenced by ntrip_text_has_space_or_control().

10.8.6.4.12 secure_clear_ntrip_credentials()

```
void secure_clear_ntrip_credentials (
            Pvt_Conf * conf)  [noexcept]
```

Scrubs the NTRIP credential fields of a Pvt_Conf copy ahead of its destruction (the Secure_String fields wipe themselves on release, but a long-lived copy should drop its secrets as soon as they are consumed). Single home of the credential field list.
References secure_clear_ntrip_credentials().
Referenced by secure_clear_ntrip_credentials().

10.9 Observables

Topics

- obs_adapters
- obs_gr_blocks
- observables_libs

10.9.1 Detailed Description

Classes for the computation of GNSS observables

10.9.2 obs_adapters

Classes

- class HybridObservables

  This class implements an ObservablesInterface for observables of all kind of GNSS signals.

10.9.2.1 Detailed Description

Wrap GNU Radio observables blocks with an ObservablesInterface

10.9.3 obs_gr_blocks

Classes

- class Gnss_circular_deque< T >
- class hybrid_observables_gs

  This class implements a block that computes observables.

Typedefs

- using hybrid_observables_gs_sptr = gnss_shared_ptr<hybrid_observables_gs>

Functions

- hybrid_observables_gs_sptr **hybrid_observables_gs_make** (const Obs_Conf &conf_)

10.9.3.1 Detailed Description

GNU Radio blocks for the computation of GNSS observables

10.9.3.2 Typedef Documentation

10.9.3.2.1 hybrid_observables_gs_sptr

```
using hybrid_observables_gs_sptr = gnss_shared_ptr<hybrid_observables_gs>
```
Definition at line 52 of file hybrid_observables_gs.h.

10.9.4 observables_libs

Classes

- class Obs_Conf

10.9.4.1 Detailed Description

Utilities for GNSS observables configuration.

10.10 Resampler

Topics

- resampler_adapters
- resampler_gr_blocks

10.10.1 Detailed Description

Classes for input signal resampling

10.10.2 resampler_adapters

Classes

- class DirectResamplerConditioner

  Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.
- class MmseResamplerConditioner

  Interface of a MMSE resampler block adapter to a SignalConditionerInterface.

10.10.2.1 Detailed Description

Classes that wrap GNU Radio resampler blocks with a GNSSBlockInterface

10.10.3 resampler_gr_blocks

Classes

- class direct_resampler_conditioner_cb

  This class implements a direct resampler conditioner for std::complex<signed char>.
- class direct_resampler_conditioner_cc

  This class implements a direct resampler conditioner for complex data.
- class direct_resampler_conditioner_cs

  This class implements a direct resampler conditioner for std::complex<short>.

Typedefs

- using direct_resampler_conditioner_cb_sptr = gnss_shared_ptr<direct_resampler_conditioner_cb>
- using direct_resampler_conditioner_cc_sptr = gnss_shared_ptr<direct_resampler_conditioner_cc>
- using direct_resampler_conditioner_cs_sptr = gnss_shared_ptr<direct_resampler_conditioner_cs>

Functions

- direct_resampler_conditioner_cb_sptr **direct_resampler_make_conditioner_cb** (double sample_freq_in, double sample_freq_out)
- direct_resampler_conditioner_cc_sptr **direct_resampler_make_conditioner_cc** (double sample_freq_in, double sample_freq_out)
- direct_resampler_conditioner_cs_sptr **direct_resampler_make_conditioner_cs** (double sample_freq_in, double sample_freq_out)

10.10.3.1 Detailed Description

GNU Radio blocks for input signal resampling

10.10.3.2 Typedef Documentation

10.10.3.2.1 direct_resampler_conditioner_cb_sptr

```
using direct_resampler_conditioner_cb_sptr = gnss_shared_ptr<direct_resampler_conditioner_cb>
```
Definition at line 35 of file direct_resampler_conditioner_cb.h.

10.10.3.2.2 direct_resampler_conditioner_cc_sptr

```
using direct_resampler_conditioner_cc_sptr = gnss_shared_ptr<direct_resampler_conditioner_cc>
```
Definition at line 40 of file direct_resampler_conditioner_cc.h.

10.10.3.2.3 direct_resampler_conditioner_cs_sptr

```
using direct_resampler_conditioner_cs_sptr = gnss_shared_ptr<direct_resampler_conditioner_cs>
```
Definition at line 34 of file direct_resampler_conditioner_cs.h.

10.11 Signal Source

Topics

- signal_source_adapters
- signal_source_gr_blocks
- signal_source_libs

10.11.1 Detailed Description

Classes for Signal Source management.

10.11.2 signal_source_adapters

Classes

- class Ad936xCustomSignalSource

  This class instantiates the Ad936xCustom gnuradio signal source. It has support also for a customized Ad936x↩ Custom firmware and signal source to support PPS samplestamp reading.
- class Adrv9361z7035SignalSourceFPGA
- class BladerfSignalSource

  This class instantiates the bladeRF signal source, streaming RX samples directly through libbladeRF (no gr-osmosdr or other intermediate driver required).

- class CustomUDPSignalSource

  This class reads from UDP packets, which streams interleaved I/Q samples over a network.
- class DMASignalSourceFPGA
- class Evk1029SignalSource

  This class reads a SAPHYRION EVK1029 raw capture file.
- class FifoSignalSource

  Class that reads a sample stream from a Unix FIFO.
- class FileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class FileSourceBase

  Base class to file-oriented SignalSourceBase GNSS blocks.
- class FileTimestampSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class FlexibandSignalSource

  This class configures and reads samples from Teleorbit Flexiband front-end. This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR).
- class Fmcomms2SignalSource
- class Fmcomms5SignalSourceFPGA
- class FourBitCpxFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class GenSignalSource

  This class wraps blocks that generates synthesized GNSS signal and filters the signal.
- class GnMaxSignalSource

  This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.
- class GSS6450FileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class IONGSMSSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class LabsatSignalSource

  This class reads samples stored in LabSat version 2, 3, and 3 Wideband format.
- class LimesdrSignalSource

  This class instantiates the LimeSDR gnuradio signal source. It has support also for a customized LimeSDR firmware and signal source to support PPS samplestamp reading.
- class MAX2771EVKITSignalSourceFPGA
- class MultichannelFileSignalSource

  Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface.
- class NsrFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class NTLabFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
- class OsmosdrSignalSource

  This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki).
- class PlutosdrSignalSource
- class PocketSdrSignalSource

  This class instantiates the gr-pocketsdr signal source for the Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends.
- class RawArraySignalSource

  This class reads samples from an antenna array RF front-end signal sampler.
- class RtlTcpSignalSource

  This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see https://osmocom.org/projects/rtl-sdr/wiki).
- class SpirFileSignalSource

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

- class TwoBitCpxFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

- class TwoBitPackedFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

- class UhdSignalSource

  This class reads samples from a UHD device (see http://code.ettus.com/redmine/ettus/projects/uhd/wiki).

- class ZmqSignalSource

  This class supports the following properties:

10.11.2.1 Detailed Description

Classes that wrap GNU Radio signal sources with a GNSSBlockInterface

10.11.3 signal_source_gr_blocks

Classes

- class ad936x_iio_source

  This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.

- class bladerf_source

  This class implements a GNU Radio source block that streams RX samples from a bladeRF device (single RX channel) using libbladeRF's synchronous streaming API, converting the device's native BLADERF_FORMAT_SC16_Q11 samples to gr_complex.

- class Evk1029Source

  Reads a continuous, header-less EVK1029 raw capture file and unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first).

- class FifoReader
- class Gr_Complex_Ip_Packet_Source
- class IONGSMSFileSource
- class labsat23_source

  This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.

- class rtl_tcp_signal_source_c

  This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types.

- class unpack_2bit_samples

  This class takes 2 bit samples that have been packed into bytes or shorts as input and generates a byte for each sample. It generates eight times as much data as is input (every two bits become 16 bits).

- class unpack_byte_2bit_cpx_samples

  This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples.

- class unpack_byte_2bit_samples

  This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.

- class unpack_byte_4bit_samples

  This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples.

- class unpack_gss6450_samples
- class unpack_intspir_1bit_samples

  This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.

- class unpack_ntlab_2bit_samples

  This class implements conversion between byte packet multichannel samples to 2bit samples 1 byte = 4 2bit samples.

- class unpack_short_byte_samples

  This class implements conversion between short packet samples to byte samples 1 short = 2 byte samples.

Typedefs

- using ad936x_iio_source_sptr = gnss_shared_ptr<ad936x_iio_source>
- using bladerf_source_sptr = gnss_shared_ptr<bladerf_source>
- using Evk1029Source_sptr = gnss_shared_ptr<Evk1029Source>
- using labsat23_source_sptr = gnss_shared_ptr<labsat23_source>
- using rtl_tcp_signal_source_c_sptr = gnss_shared_ptr<rtl_tcp_signal_source_c>
- using b_io_context = boost::asio::io_service
- using unpack_2bit_samples_sptr = gnss_shared_ptr<unpack_2bit_samples>
- using unpack_byte_2bit_cpx_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_cpx_samples>
- using unpack_byte_2bit_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_samples>
- using unpack_byte_4bit_samples_sptr = gnss_shared_ptr<unpack_byte_4bit_samples>
- using unpack_gss6450_samples_sptr = gnss_shared_ptr<unpack_gss6450_samples>
- using unpack_intspir_1bit_samples_sptr = gnss_shared_ptr<unpack_intspir_1bit_samples>
- using unpack_ntlab_2bit_samples_sptr = gnss_shared_ptr<unpack_ntlab_2bit_samples>
- using unpack_short_byte_samples_sptr = gnss_shared_ptr<unpack_short_byte_samples>

Functions

- ad936x_iio_source_sptr **ad936x_iio_make_source_sptr** (const std::string &pluto_uri_, const std::string &board_type_, int64_t bandwidth_, int64_t sample_rate_, int64_t freq_, const std::string &rf_port_select_, const std::string &rf_filter, const std::string &gain_mode_rx0_, const std::string &gain_mode_rx1_, double rf_gain_rx0_, double rf_gain_rx1_, bool enable_ch0, bool enable_ch1, int64_t freq_2ch, bool ppsmode_, bool customsamplesize_, const std::string &fe_ip_, int fe_ctlport_, int ssize_, int bshift_, bool spattern_, double lo_attenuation_db_, bool high_side_lo_, int tx_lo_channel_)
- bladerf_source_sptr **bladerf_make_source_sptr** (const std::string &bladerf_args, const std::string &device_serial, double sample_rate, double freq, double bandwidth, double gain, bool agc_enabled, bool rx_bias_tee, unsigned int num_buffers, unsigned int buffer_size, unsigned int num_transfers, unsigned int stream_timeout_ms)
- Evk1029Source_sptr **evk1029_make_source** (const std::string &filename, Concurrent_Queue< pmt::pmt_t > *queue, double sampling_frequency, uint64_t bytes_to_skip=0)
- labsat23_source_sptr **labsat23_make_source_sptr** (const char *signal_file_basename, const std::vector< int > &channel_selector, Concurrent_Queue< pmt::pmt_t > *queue, bool digital_io_enabled, double seconds_to_skip)
- rtl_tcp_signal_source_c_sptr **rtl_tcp_make_signal_source_c** (const std::string &address, int16_t port, bool flip_iq=false)
- unpack_2bit_samples_sptr **make_unpack_2bit_samples** (bool big_endian_bytes, size_t item_size, bool big_endian_items, bool reverse_interleaving=false)
- unpack_byte_2bit_cpx_samples_sptr **make_unpack_byte_2bit_cpx_samples** ()
- unpack_byte_2bit_samples_sptr **make_unpack_byte_2bit_samples** ()
- unpack_byte_4bit_samples_sptr **make_unpack_byte_4bit_samples** ()
- unpack_gss6450_samples_sptr **make_unpack_gss6450_samples** (int adc_nbit_, bool gss6425_compatible=false)
- unpack_intspir_1bit_samples_sptr **make_unpack_intspir_1bit_samples** ()
- unpack_ntlab_2bit_samples_sptr **make_unpack_ntlab_2bit_samples** (size_t item_size, int nchannels=4)
- unpack_short_byte_samples_sptr **make_unpack_short_byte_samples** ()

10.11.3.1 Detailed Description

GNU Radio blocks for signal sources.

10.11.3.2 Typedef Documentation

10.11.3.2.1 ad936x_iio_source_sptr

```
using ad936x_iio_source_sptr = gnss_shared_ptr<ad936x_iio_source>
```

Definition at line 45 of file ad936x_iio_source.h.

10.11.3.2.2 b_io_context

```
using b_io_context = boost::asio::io_service
```
Definition at line 53 of file rtl_tcp_signal_source_c.h.

10.11.3.2.3 bladerf_source_sptr

```
using bladerf_source_sptr = gnss_shared_ptr<bladerf_source>
```
Definition at line 37 of file bladerf_source.h.

10.11.3.2.4 Evk1029Source_sptr

```
using Evk1029Source_sptr = gnss_shared_ptr<Evk1029Source>
```
Definition at line 60 of file evk1029_source.h.

10.11.3.2.5 labsat23_source_sptr

```
using labsat23_source_sptr = gnss_shared_ptr<labsat23_source>
```
Definition at line 44 of file labsat23_source.h.

10.11.3.2.6 rtl_tcp_signal_source_c_sptr

```
using rtl_tcp_signal_source_c_sptr = gnss_shared_ptr<rtl_tcp_signal_source_c>
```
Definition at line 48 of file rtl_tcp_signal_source_c.h.

10.11.3.2.7 unpack_2bit_samples_sptr

```
using unpack_2bit_samples_sptr = gnss_shared_ptr<unpack_2bit_samples>
```
Definition at line 71 of file unpack_2bit_samples.h.

10.11.3.2.8 unpack_byte_2bit_cpx_samples_sptr

```
using unpack_byte_2bit_cpx_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_cpx_samples>
```
Definition at line 36 of file unpack_byte_2bit_cpx_samples.h.

10.11.3.2.9 unpack_byte_2bit_samples_sptr

```
using unpack_byte_2bit_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_samples>
```
Definition at line 32 of file unpack_byte_2bit_samples.h.

10.11.3.2.10 unpack_byte_4bit_samples_sptr

```
using unpack_byte_4bit_samples_sptr = gnss_shared_ptr<unpack_byte_4bit_samples>
```
Definition at line 34 of file unpack_byte_4bit_samples.h.

10.11.3.2.11 unpack_gss6450_samples_sptr

```
using unpack_gss6450_samples_sptr = gnss_shared_ptr<unpack_gss6450_samples>
```
Definition at line 34 of file unpack_gss6450_samples.h.

10.11.3.2.12 unpack_intspir_1bit_samples_sptr

```
using unpack_intspir_1bit_samples_sptr = gnss_shared_ptr<unpack_intspir_1bit_samples>
```
Definition at line 33 of file unpack_intspir_1bit_samples.h.

10.11.3.2.13 unpack_ntlab_2bit_samples_sptr

```
using unpack_ntlab_2bit_samples_sptr = gnss_shared_ptr<unpack_ntlab_2bit_samples>
```
Definition at line 35 of file unpack_ntlab_2bit_samples.h.

10.11.3.2.14 unpack_short_byte_samples_sptr

```
using unpack_short_byte_samples_sptr = gnss_shared_ptr<unpack_short_byte_samples>
```

Definition at line 33 of file unpack_short_byte_samples.h.

10.11.4 signal_source_libs

Classes

- struct stream_cfg
- class ad936x_iio_custom
- class ad936x_iio_samples
- class Fpga_buffer_monitor

  Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers.
- class Fpga_dynamic_bit_selection

  Controls dynamic bit selection in the FPGA.
- class Fpga_Switch

  Class that controls the switch in the FPGA, which connects the FPGA acquisition and multicorrelator modules to either the DMA or the Analog Front-End.
- class Gnss_Sdr_Timestamp
- class Gnss_Sdr_Valve

  Implementation of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.
- struct IONGSMSChunkUnpackingCtx< WT >
- class Rtl_Tcp_Dongle_Info

  This class represents the dongle information which is sent by rtl_tcp.

Macros

- #define FIR_BUF_SIZE 8192

Enumerations

- enum **iodev** { **RX** , **TX** }
- enum RTL_TCP_COMMAND {
  RTL_TCP_SET_FREQUENCY = 1 , **RTL_TCP_SET_SAMPLE_RATE** = 2 , **RTL_TCP_SET_GAIN_MODE** = 3 , **RTL_TCP_SET_GAIN** = 4 ,
  RTL_TCP_SET_IF_GAIN = 6 , **RTL_TCP_SET_AGC_MODE** = 8 }

  Command IDs for configuration rtl_tcp.

Functions

- void **errchk** (int v, const char ∗what)
- void **wr_ch_lli** (struct iio_channel ∗chn, const char ∗what, int64_t val)
- void **wr_ch_str** (struct iio_channel ∗chn, const char ∗what, const char ∗str)
- struct iio_device ∗ **get_ad9361_phy** (struct iio_context ∗ctx)
- bool **get_ad9361_stream_dev** (struct iio_context ∗ctx, enum iodev d, struct iio_device ∗∗dev)
- bool **get_ad9361_stream_ch** (struct iio_context ∗ctx, enum iodev d, struct iio_device ∗dev, int chid, struct iio_channel ∗∗chn)
- bool **get_phy_chan** (struct iio_context ∗ctx, enum iodev d, int chid, struct iio_channel ∗∗chn)
- bool **get_lo_chan** (struct iio_context ∗ctx, enum iodev d, struct iio_channel ∗∗chn)
- bool **cfg_ad9361_streaming_ch** (struct iio_context ∗ctx, struct stream_cfg ∗cfg, enum iodev type, int chid)
- bool **config_ad9361_rx_local** (uint64_t bandwidth_, uint64_t sample_rate_, uint64_t freq0_, uint64_t freq1_, const std::string &rf_port_select_, bool rx1_enable_, bool rx2_enable_, const std::string &gain_mode_rx1_, const std::string &gain_mode_rx2_, double rf_gain_rx1_, double rf_gain_rx2_, bool quadrature_, bool rfdc_, bool bbdc_, std::string filter_source_, std::string filter_filename_, float Fpass_, float Fstop_)

- bool **config_ad9361_rx_remote** (const std::string &remote_host, uint64_t bandwidth_, uint64_t sample_↩ rate_, uint64_t freq_, const std::string &rf_port_select_, bool rx1_enable_, bool rx2_enable_, const std::↩ string &gain_mode_rx1_, const std::string &gain_mode_rx2_, double rf_gain_rx1_, double rf_gain_rx2_, bool quadrature_, bool rfdc_, bool bbdc_, std::string filter_source_, std::string filter_filename_, float Fpass_, float Fstop_)
- bool **config_ad9361_lo_local** (uint64_t bandwidth_, uint64_t sample_rate_, uint64_t freq_rf_tx_hz_, double tx_attenuation_db_, int64_t freq_dds_tx_hz_, double scale_dds_dbfs_, double phase_dds_deg_)
- bool **config_ad9361_lo_remote** (const std::string &remote_host, uint64_t bandwidth_, uint64_t sample_↩ rate_, uint64_t freq_rf_tx_hz_, double tx_attenuation_db_, int64_t freq_dds_tx_hz_, double scale_dds_dbfs_, double phase_dds_deg_)
- bool **ad9361_disable_lo_remote** (const std::string &remote_host)
- bool **ad9361_disable_lo_local** ()
- bool **load_fir_filter** (std::string &filter, struct iio_device ∗phy)
- bool **disable_ad9361_rx_local** ()
- bool **disable_ad9361_rx_remote** (const std::string &remote_host)
- gnss_shared_ptr< Gnss_Sdr_Timestamp > **gnss_sdr_make_Timestamp** (size_t sizeof_stream_item, std::string timestamp_file, double clock_offset_ms, int items_to_samples)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > ∗queue)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > ∗queue, bool stop_flowgraph)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > ∗queue, bool stop_flowgraph, int n_streams)
- boost::system::error_code rtl_tcp_command (RTL_TCP_COMMAND id, unsigned param, boost::asio::ip::↩ tcp::socket &socket)

  Send a command to rtl_tcp over the given socket.

10.11.4.1 Detailed Description

Library with utilities for signal sources.

10.11.4.2 Macro Definition Documentation

10.11.4.2.1 FIR_BUF_SIZE

```
#define FIR_BUF_SIZE 8192
```

Definition at line 33 of file ad9361_manager.h.

10.11.4.3 Enumeration Type Documentation

10.11.4.3.1 iodev

```
enum iodev
```

Definition at line 43 of file ad9361_manager.h.

10.11.4.3.2 RTL_TCP_COMMAND

```
enum RTL_TCP_COMMAND
```

Command IDs for configuration rtl_tcp.
Definition at line 32 of file rtl_tcp_commands.h.

10.11.4.4 Function Documentation

10.11.4.4.1 rtl_tcp_command()

```
boost::system::error_code rtl_tcp_command (
            RTL_TCP_COMMAND id,
            unsigned param,
            boost::asio::ip::tcp::socket & socket)
```

Send a command to rtl_tcp over the given socket.
References rtl_tcp_command().
Referenced by rtl_tcp_command().

10.12 Telemetry_Decoder

Topics

- Telemetry_Decoder_adapters
- telemetry_decoder_gr_blocks
- telemetry_decoder_libswiftcnav
- telemetry_decoder_libs

10.12.1 Detailed Description

10.12.2 Telemetry_Decoder_adapters

Classes

- class TelemetryDecoderAdapter

  Base class for Telemetry Decoder adapters.

10.12.2.1 Detailed Description

10.12.3 telemetry_decoder_gr_blocks

Classes

- class beidou_dnav_telemetry_decoder_gs

  This class implements a block that decodes the BeiDou DNAV data, broadcast by both the B1I and B3I signals (frequency_band = 1 or 3, respectively).
- class galileo_telemetry_decoder_gs

  This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.
- class glonass_gnav_telemetry_decoder_gs

  This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals (frequency_band = 1 or 2, respectively).
- class gps_l1_ca_telemetry_decoder_gs

  This class implements a block that decodes the NAV data defined in IS-GPS-200M.
- class gps_l2c_telemetry_decoder_gs

  This class implements a block that decodes CNAV data defined in IS-GPS-200M.
- class gps_l5_telemetry_decoder_gs

  This class implements a GPS L5 Telemetry decoder.
- class SbasL1DumpSession
- class sbas_l1_telemetry_decoder_gs

  This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229.
- class telemetry_impl_interface

  Common base class for telemetry decoder GNU Radio implementations.

Typedefs

- using beidou_dnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_dnav_telemetry_decoder_gs>
- using galileo_telemetry_decoder_gs_sptr = gnss_shared_ptr<galileo_telemetry_decoder_gs>
- using glonass_gnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<glonass_gnav_telemetry_decoder_gs>
- using gps_l1_ca_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l1_ca_telemetry_decoder_gs>
- using gps_l2c_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l2c_telemetry_decoder_gs>
- using gps_l5_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l5_telemetry_decoder_gs>
- using sbas_l1_telemetry_decoder_gs_sptr = gnss_shared_ptr<sbas_l1_telemetry_decoder_gs>
- using telemetry_impl_interface_sptr = gnss_shared_ptr<telemetry_impl_interface>

Enumerations

- enum class **L1LnavSystem** { **GPS** , **QZSS** }

Functions

- beidou_dnav_telemetry_decoder_gs_sptr **beidou_dnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)
- galileo_telemetry_decoder_gs_sptr **galileo_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int frame_type)
- glonass_gnav_telemetry_decoder_gs_sptr **glonass_gnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)
- gps_l1_ca_telemetry_decoder_gs_sptr **gps_l1_ca_make_telemetry_decoder_gs** (const Tlm_Conf &conf, L1LnavSystem system=L1LnavSystem::GPS)
- gps_l2c_telemetry_decoder_gs_sptr **gps_l2c_make_telemetry_decoder_gs** (const Tlm_Conf &conf)
- gps_l5_telemetry_decoder_gs_sptr **gps_l5_make_telemetry_decoder_gs** (const Tlm_Conf &conf, Cnav↩ System system=CnavSystem::GPS)
- sbas_l1_telemetry_decoder_gs_sptr **sbas_l1_make_telemetry_decoder_gs** (bool dump, std::string dump_filename=std::string())

10.12.3.1 Detailed Description

GNU Radio blocks for the demodulation of GNSS navigation messages.

10.12.3.2 Typedef Documentation

10.12.3.2.1 beidou_dnav_telemetry_decoder_gs_sptr

```
using beidou_dnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_dnav_telemetry_decoder_gs>
```

Definition at line 40 of file beidou_dnav_telemetry_decoder_gs.h.

10.12.3.2.2 galileo_telemetry_decoder_gs_sptr

```
using galileo_telemetry_decoder_gs_sptr = gnss_shared_ptr<galileo_telemetry_decoder_gs>
```

Definition at line 42 of file galileo_telemetry_decoder_gs.h.

10.12.3.2.3 glonass_gnav_telemetry_decoder_gs_sptr

```
using glonass_gnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<glonass_gnav_telemetry_decoder_gs>
```

Definition at line 39 of file glonass_gnav_telemetry_decoder_gs.h.

10.12.3.2.4 gps_l1_ca_telemetry_decoder_gs_sptr

```
using gps_l1_ca_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l1_ca_telemetry_decoder_gs>
```

Definition at line 44 of file gps_l1_ca_telemetry_decoder_gs.h.

10.12.3.2.5 gps_l2c_telemetry_decoder_gs_sptr

```
using gps_l2c_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l2c_telemetry_decoder_gs>
```

Definition at line 39 of file gps_l2c_telemetry_decoder_gs.h.

10.12.3.2.6 gps_l5_telemetry_decoder_gs_sptr

```
using gps_l5_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l5_telemetry_decoder_gs>
```

Definition at line 40 of file gps_l5_telemetry_decoder_gs.h.

10.12.3.2.7 sbas_l1_telemetry_decoder_gs_sptr

```
using sbas_l1_telemetry_decoder_gs_sptr = gnss_shared_ptr<sbas_l1_telemetry_decoder_gs>
```

Definition at line 60 of file sbas_l1_telemetry_decoder_gs.h.

10.12.3.2.8 telemetry_impl_interface_sptr

```
using telemetry_impl_interface_sptr = gnss_shared_ptr<telemetry_impl_interface>
```
Definition at line 41 of file telemetry_impl_interface.h.

10.12.3.3 Enumeration Type Documentation

10.12.3.3.1 L1LnavSystem

```
enum class L1LnavSystem  [strong]
```
Definition at line 35 of file gps_l1_ca_telemetry_decoder_gs.h.

10.12.4 telemetry_decoder_libswiftcnav

Classes

- struct cnav_msg_t
- struct cnav_v27_part_t
- struct cnav_msg_decoder_t

Macros

- #define GPS_L2_V27_HISTORY_LENGTH_BITS 64
- #define GPS_L2C_V27_INIT_BITS (32)
- #define GPS_L2C_V27_DECODE_BITS (32)
- #define GPS_L2C_V27_DELAY_BITS (32)
- #define ABS(x)
- #define MIN(x, y)
- #define MAX(x, y)
- #define CLAMP_DIFF(a, b)

Functions

- uint8_t **parity** (uint32_t x)
- uint32_t **getbitu** (const uint8_t *buff, uint32_t pos, uint8_t len)
- int32_t **getbits** (const uint8_t *buff, uint32_t pos, uint8_t len)
- void **setbitu** (uint8_t *buff, uint32_t pos, uint32_t len, uint32_t data)
- void **setbits** (uint8_t *buff, uint32_t pos, uint32_t len, int32_t data)
- void **bitcopy** (void *dst, uint32_t dst_index, const void *src, uint32_t src_index, uint32_t count)
- void **bitshl** (void *buf, uint32_t size, uint32_t shift)
- uint8_t **count_bits_u64** (uint64_t v, uint8_t bv)
- uint8_t **count_bits_u32** (uint32_t v, uint8_t bv)
- uint8_t **count_bits_u16** (uint16_t v, uint8_t bv)
- uint8_t **count_bits_u8** (uint8_t v, uint8_t bv)
- const v27_poly_t * **cnav_msg_decoder_get_poly** (void)
- void **cnav_msg_decoder_init** (cnav_msg_decoder_t *dec)
- bool **cnav_msg_decoder_add_symbol** (cnav_msg_decoder_t *dec, unsigned char symbol, cnav_msg_t *msg, uint32_t *delay)
- uint32_t **crc24q** (const uint8_t *buf, uint32_t len, uint32_t crc)
- uint32_t **crc24q_bits** (uint32_t crc, const uint8_t *buf, uint32_t n_bits, bool invert)

10.12.4.1 Detailed Description

Utilities for CNAV message decoding by Swift Navigation Inc.

10.12.4.2 Macro Definition Documentation

10.12.4.2.1 ABS

```
#define ABS(
              x)
```

Value:

```
((x) < 0 ? -(x) : (x))
```

Definition at line 34 of file swift_common.h.

10.12.4.2.2 CLAMP_DIFF

```
#define CLAMP_DIFF(
              a,
              b)
```

Value:

```
(MAX((a), (b)) - (b))
```

Definition at line 37 of file swift_common.h.

10.12.4.2.3 GPS_L2_V27_HISTORY_LENGTH_BITS

```
#define GPS_L2_V27_HISTORY_LENGTH_BITS 64
```

Size of the Viterbi decoder history.
Definition at line 40 of file cnav_msg.h.

10.12.4.2.4 GPS_L2C_V27_DECODE_BITS

```
#define GPS_L2C_V27_DECODE_BITS (32)
```

Bits to decode at a time.
Definition at line 44 of file cnav_msg.h.

10.12.4.2.5 GPS_L2C_V27_DELAY_BITS

```
#define GPS_L2C_V27_DELAY_BITS (32)
```

Bits in decoder tail. We ignore them.
Definition at line 46 of file cnav_msg.h.

10.12.4.2.6 GPS_L2C_V27_INIT_BITS

```
#define GPS_L2C_V27_INIT_BITS (32)
```

Bits to accumulate before decoding starts.
Definition at line 42 of file cnav_msg.h.

10.12.4.2.7 MAX

```
#define MAX(
              x,
              y)
```

Value:

```
(((x) > (y)) ? (x) : (y))
```

Definition at line 36 of file swift_common.h.

10.12.4.2.8 MIN

```
#define MIN(
              x,
              y)
```

Value:

```
(((x) < (y)) ? (x) : (y))
```

Definition at line 35 of file swift_common.h.

10.12.5 telemetry_decoder_libs

Classes

- class Tlm_Conf
- class Tlm_CRC_Stats

  Class that computes the telemetry CRC statistics.
- class Viterbi_Decoder

  Class that implements a Viterbi decoder.
- class Viterbi_Decoder_Sbas

  Class that implements a Viterbi decoder.
- class Serdes_Nav_Message

  This class implements serialization and deserialization of Nav_Message_Packet objects using Protocol Buffers.

Functions

- int **save_tlm_matfile** (const std::string &dumpfile)
- bool **tlm_remove_file** (const std::string &file_to_remove)
- void **tlm_cleanup_and_save_files** (std::ofstream &dump_file, const std::string &dump_filename, bool dump, bool dump_mat, bool remove_dat)

10.12.5.1 Detailed Description

Utilities for the decoding of GNSS navigation messages.

10.13 Tracking

Topics

- tracking_adapters
- tracking_gr_blocks
- tracking_libs

10.13.1 Detailed Description

Classes for GNSS signal tracking.

10.13.2 tracking_adapters

Classes

- class DllPllTrackingAdapter

  Base class providing shared logic for DLL+PLL tracking loop adapters for GNSS signals.
- class GalileoE1TcpConnectorTracking

  This class implements a code DLL + carrier PLL tracking loop.
- class GpsL1CaDllPllTrackingGPU

  This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions.
- class GpsL1CaGaussianTracking

  This class implements a code DLL + carrier PLL tracking loop.
- class GpsL1CaTcpConnectorTracking

  This class implements a code DLL + carrier PLL tracking loop.

10.13.2.1 Detailed Description

Wrap GNU Radio blocks for GNSS signal tracking with a TrackingInterface

10.13.3 tracking_gr_blocks

Classes

- class dll_pll_veml_tracking

  This class implements a code DLL + carrier PLL tracking block.
- class dll_pll_veml_tracking_fpga

  This class implements a code DLL + carrier PLL tracking block.
- class Galileo_E1_Tcp_Connector_Tracking_cc

  This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.
- class Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc

  This class implements a DLL + PLL tracking loop block.
- class Gps_L1_Ca_Gaussian_Tracking_cc

  This class implements a DLL + PLL tracking loop block.
- class Gps_L1_Ca_Tcp_Connector_Tracking_cc

  This class implements a DLL + PLL tracking loop block.

Typedefs

- using dll_pll_veml_tracking_sptr = gnss_shared_ptr<dll_pll_veml_tracking>
- using dll_pll_veml_tracking_fpga_sptr = gnss_shared_ptr<dll_pll_veml_tracking_fpga>
- using galileo_e1_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Galileo_E1_Tcp_Connector_Tracking_cc>
- using gps_l1_ca_dll_pll_tracking_gpu_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc>
- using gps_l1_ca_gaussian_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Gaussian_Tracking_cc>
- using gps_l1_ca_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Tcp_Connector_Tracking_cc>

Functions

- dll_pll_veml_tracking_sptr **dll_pll_veml_make_tracking** (const Dll_Pll_Conf &conf_)
- dll_pll_veml_tracking_fpga_sptr **dll_pll_veml_make_tracking_fpga** (const Dll_Pll_Conf_Fpga &conf_)
- galileo_e1_tcp_connector_tracking_cc_sptr **galileo_e1_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float early_late_space_chips, float very_early_late_space_chips, size_t port_ch0)
- gps_l1_ca_dll_pll_tracking_gpu_cc_sptr **gps_l1_ca_dll_pll_make_tracking_gpu_cc** (int64_t fs_in, uint32_t vector_length, bool dump, std::string dump_filename, float pll_bw_hz, float dll_bw_hz, float early_late_space_chips)
- gps_l1_ca_gaussian_tracking_cc_sptr **gps_l1_ca_gaussian_make_tracking_cc** (uint32_t order, int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float dll_bw_hz, float early_late_space_chips, bool bce_run, uint32_t bce_ptrans, uint32_t bce_strans, int32_t bce_nu, int32_t bce_kappa)
- gps_l1_ca_tcp_connector_tracking_cc_sptr **gps_l1_ca_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float early_late_space_chips, size_t port_ch0)

10.13.3.1 Detailed Description

GNU Radio blocks for GNSS signal tracking.

10.13.3.2 Typedef Documentation

10.13.3.2.1 dll_pll_veml_tracking_fpga_sptr

```
using dll_pll_veml_tracking_fpga_sptr = gnss_shared_ptr<dll_pll_veml_tracking_fpga>
```

Definition at line 51 of file dll_pll_veml_tracking_fpga.h.

10.13.3.2.2 dll_pll_veml_tracking_sptr

```
using dll_pll_veml_tracking_sptr = gnss_shared_ptr<dll_pll_veml_tracking>
```

Definition at line 55 of file dll_pll_veml_tracking.h.

10.13.3.2.3 galileo_e1_tcp_connector_tracking_cc_sptr

```
using galileo_e1_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Galileo_E1_Tcp_Connector_Tracking_cc>
```
Definition at line 46 of file galileo_e1_tcp_connector_tracking_cc.h.

10.13.3.2.4 gps_l1_ca_dll_pll_tracking_gpu_cc_sptr

```
using gps_l1_ca_dll_pll_tracking_gpu_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc>
```
Definition at line 44 of file gps_l1_ca_dll_pll_tracking_gpu_cc.h.

10.13.3.2.5 gps_l1_ca_gaussian_tracking_cc_sptr

```
using gps_l1_ca_gaussian_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Gaussian_Tracking_cc>
```
Definition at line 54 of file gps_l1_ca_gaussian_tracking_cc.h.

10.13.3.2.6 gps_l1_ca_tcp_connector_tracking_cc_sptr

```
using gps_l1_ca_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Tcp_Connector_Tracking_cc>
```
Definition at line 44 of file gps_l1_ca_tcp_connector_tracking_cc.h.

10.13.4 tracking_libs

Classes

- class Bayesian_estimator

  Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance).
- class HistogramBitSynchronizer

  Histogram-based navigation data bit synchronizer.
- class Cpu_Multicorrelator

  Class that implements carrier wipe-off and correlators.
- class Cpu_Multicorrelator_16sc

  Class that implements carrier wipe-off and correlators.
- class Cpu_Multicorrelator_Real_Codes

  Class that implements carrier wipe-off and correlators.
- struct GPU_Complex
- struct GPU_Complex_Short
- class cuda_multicorrelator

  Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators.
- class Dll_Pll_Conf
- class Dll_Pll_Conf_Fpga
- class Exponential_Smoother

  Class that implements a first-order exponential smoother.
- class Fpga_Multicorrelator_8sc

  Class that implements carrier wipe-off and correlators.
- class ModelFunction
- class CubatureFilter
- class UnscentedFilter
- class Tcp_Communication

  TCP communication class.
- class Tcp_Packet_Data

  Class that implements a TCP data packet.
- class Tracking_2nd_DLL_filter

  This class implements a 2nd order DLL filter for code tracking loop.
- class Tracking_2nd_PLL_filter

  This class implements a 2nd order PLL filter for carrier tracking loop.

- class Tracking_FLL_PLL_filter

  This class implements a hybrid FLL and PLL filter for tracking carrier loop.
- class Tracking_loop_filter

  This class implements a generic 1st, 2nd or 3rd order loop filter.

Macros

- #define NUM_TX_VARIABLES_GALILEO_E1 13
- #define NUM_TX_VARIABLES_GPS_L1_CA 9
- #define NUM_RX_VARIABLES 4

Typedefs

- using b_io_context = boost::asio::io_service

Functions

- float cn0_svn_estimator (const gr_complex ∗Prompt_buffer, int length, float coh_integration_time_s)

  cn0_svn_estimator is a Carrier-to-Noise (CN0) estimator based on the Signal-to-Noise Variance (SNV) estimator
- float cn0_m2m4_estimator (const gr_complex ∗Prompt_buffer, int length, float coh_integration_time_s)

  cn0_m2m4_estimator is a Carrier-to-Noise (CN0) estimator based on the Second- and Fourth-Order Moments Method (M2M4)
- float carrier_lock_detector (const gr_complex ∗Prompt_buffer, int length)

  A carrier lock detector.
- double fll_four_quadrant_atan (gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2)
- double **fll_diff_atan** (gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2)
- double phase_unwrap (double phase_rad)

  Phase unwrapping function, input is [rad].
- double pll_four_quadrant_atan (gr_complex prompt_s1)

  PLL four quadrant arctan discriminator.
- double pll_cloop_two_quadrant_atan (gr_complex prompt_s1)

  PLL Costas loop two quadrant arctan discriminator.
- double dll_nc_e_minus_l_normalized (gr_complex early_s1, gr_complex late_s1, float spc=0.5, float slope=1.0, float y_intercept=1.0)

  DLL Noncoherent Early minus Late envelope normalized discriminator.
- double dll_nc_vemlp_normalized (gr_complex very_early_s1, gr_complex early_s1, gr_complex late_s1, gr↩ _complex very_late_s1)

  DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator.
- template<typename Fun>
  double CalculateSlope (Fun &&f, double x)
- template<typename Fun>
  double CalculateSlopeAbs (Fun &&f, double x)
- template<typename Fun>
  double GetYIntercept (Fun &&f, double x)
- template<typename Fun>
  double GetYInterceptAbs (Fun &&f, double x)
- template<int M = 1, int N = M>
  double SinBocCorrelationFunction (double offset_in_chips)
- template<int M = 1, int N = M>
  double CosBocCorrelationFunction (double offset_in_chips)

10.13.4.1 Detailed Description

Utitilies for GNSS signal tracking.

10.13.4.2 Macro Definition Documentation

10.13.4.2.1 NUM_RX_VARIABLES

```
#define NUM_RX_VARIABLES 4
```
Definition at line 39 of file tcp_communication.h.

10.13.4.2.2 NUM_TX_VARIABLES_GALILEO_E1

```
#define NUM_TX_VARIABLES_GALILEO_E1 13
```
Definition at line 37 of file tcp_communication.h.

10.13.4.2.3 NUM_TX_VARIABLES_GPS_L1_CA

```
#define NUM_TX_VARIABLES_GPS_L1_CA 9
```
Definition at line 38 of file tcp_communication.h.

10.13.4.3 Typedef Documentation

10.13.4.3.1 b_io_context

```
using b_io_context = boost::asio::io_service
```
Definition at line 34 of file tcp_communication.h.

10.13.4.4 Function Documentation

10.13.4.4.1 CalculateSlope()

```
template<typename Fun>
double CalculateSlope (
            Fun && f,
            double x)
```
Definition at line 112 of file tracking_discriminators.h.

10.13.4.4.2 CalculateSlopeAbs()

```
template<typename Fun>
double CalculateSlopeAbs (
            Fun && f,
            double x)
```
Definition at line 120 of file tracking_discriminators.h.

10.13.4.4.3 carrier_lock_detector()

```
float carrier_lock_detector (
            const gr_complex * Prompt_buffer,
            int length)
```
A carrier lock detector.
The Carrier Phase Lock Detector block uses the estimate of the cosine of twice the carrier phase error is given by

$$C2\phi = \frac{NBD}{NBP}, \tag{10.1}$$

where $NBD = (\sum_{i=0}^{N-1} |Im(Pc(i))|)^2 + (\sum_{i=0}^{N-1} |Re(Pc(i))|)^2$, $NBP = \sum_{i=0}^{N-1} Im(Pc(i))^2 - \sum_{i=0}^{N-1} Re(Pc(i))^2$, and $Pc(i)$ is the prompt correlator output for the sample index i. Ref: Van Dierendonck, A.J. (1996), Global Positioning System: Theory and Applications, Volume I, Chapter 8: GPS Receivers, AJ Systems, Los Altos, CA 94024. Inc.: 329-407.
References carrier_lock_detector().
Referenced by carrier_lock_detector().

10.13.4.4.4 cn0_m2m4_estimator()

```
float cn0_m2m4_estimator (
            const gr_complex * Prompt_buffer,
            int length,
            float coh_integration_time_s)
```

cn0_m2m4_estimator is a Carrier-to-Noise (CN0) estimator based on the Second- and Fourth-Order Moments Method (M2M4)
Signal-to-Noise (SNR) (ρ) estimator using the Moments Method:

$$\hat{\rho} = \frac{\sqrt{2\hat{M}_2^2 - \hat{M}_4}}{\hat{M}_2 - \sqrt{2\hat{M}_2^2 - \hat{M}_4}}, \tag{10.2}$$

where $\hat{M}_2 = \frac{1}{N}\sum_{k=0}^{K-1}|P[k]|^2$, $\hat{M}_4 = \frac{1}{K}\sum_{k=0}^{K-1}|P[k]|^4$, $|\cdot|$ is the absolute value, and $P[k]$ is the prompt correlator output for the sample index k.
The SNR value is converted to CN0 [dB-Hz] taking into account the coherent integration time, using the following formula:

$$CN0_{dB} = 10 * log(\hat{\rho}) - 10 * log(T_{int}), \tag{10.3}$$

where T_{int} is the coherent integration time, in seconds.
Ref: D. R. Pauluzzi, N. C. Beaulieu, "A comparison of SNR estimation techniques for the AWGN channel," IEEE Trans. on Comm., vol. 48, no. 10, pp. 1681–1691, Oct. 2000.
References cn0_m2m4_estimator().
Referenced by cn0_m2m4_estimator().

10.13.4.4.5 cn0_svn_estimator()

```
float cn0_svn_estimator (
            const gr_complex * Prompt_buffer,
            int length,
            float coh_integration_time_s)
```

cn0_svn_estimator is a Carrier-to-Noise (CN0) estimator based on the Signal-to-Noise Variance (SNV) estimator
Signal-to-Noise (SNR) (ρ) estimator using the Signal-to-Noise Variance (SNV) estimator:

$$\hat{\rho} = \frac{\hat{P}_s}{\hat{P}_n} = \frac{\hat{P}_s}{\hat{P}_{tot} - \hat{P}_s}, \tag{10.4}$$

where $\hat{P}_s = \left(\frac{1}{N}\sum_{i=0}^{N-1}|Re(Pc(i))|\right)^2$ is the estimation of the signal power, $\hat{P}_{tot} = \frac{1}{N}\sum_{i=0}^{N-1}|Pc(i)|^2$ is the estimator of the total power, $|\cdot|$ is the absolute value, $Re(\cdot)$ stands for the real part of the value, and $Pc(i)$ is the prompt correlator output for the sample index i.
The SNR value is converted to CN0 [dB-Hz], taking into account the coherent integration time, using the following formula:

$$CN0_{dB} = 10 * log(\hat{\rho}) - 10 * log(T_{int}), \tag{10.5}$$

where T_{int} is the coherent integration time, in seconds.
Ref: Marco Pini, Emanuela Falletti and Maurizio Fantino, "Performance Evaluation of C/N0 Estimators using a Real Time GNSS Software Receiver," IEEE 10th International Symposium on Spread Spectrum Techniques and Applications, pp.28-30, August 2008.
References cn0_svn_estimator().
Referenced by cn0_svn_estimator().

10.13.4.4.6 CosBocCorrelationFunction()

```
template<int M = 1, int N = M>
double CosBocCorrelationFunction (
            double offset_in_chips)
```

Definition at line 171 of file tracking_discriminators.h.

10.13.4.4.7 dll_nc_e_minus_l_normalized()

```
double dll_nc_e_minus_l_normalized (
            gr_complex early_s1,
            gr_complex late_s1,
            float spc = 0.5,
            float slope = 1.0,
            float y_intercept = 1.0)
```

DLL Noncoherent Early minus Late envelope normalized discriminator.
DLL Noncoherent Early minus Late envelope normalized discriminator:

$$error = \frac{y_{intercept} - \text{slope} * \epsilon}{\text{slope}} \frac{E - L}{E + L}, \tag{10.6}$$

where $E = \sqrt{I_{ES}^2 + Q_{ES}^2}$ is the Early correlator output absolute value and $L = \sqrt{I_{LS}^2 + Q_{LS}^2}$ is the Late correlator output absolute value. The output is in [chips].
References dll_nc_e_minus_l_normalized().
Referenced by dll_nc_e_minus_l_normalized().

10.13.4.4.8 dll_nc_vemlp_normalized()

```
double dll_nc_vemlp_normalized (
            gr_complex very_early_s1,
            gr_complex early_s1,
            gr_complex late_s1,
            gr_complex very_late_s1)
```

DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator.
DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator, using the outputs of four correlators, Very Early (VE), Early (E), Late (L) and Very Late (VL):

$$error = \frac{E - L}{E + L}, \tag{10.7}$$

where $E = \sqrt{I_{VE}^2 + Q_{VE}^2 + I_E^2 + Q_E^2}$ and $L = \sqrt{I_{VL}^2 + Q_{VL}^2 + I_L^2 + Q_L^2}$. The output is in [chips].
References dll_nc_vemlp_normalized().
Referenced by dll_nc_vemlp_normalized().

10.13.4.4.9 fll_four_quadrant_atan()

```
double fll_four_quadrant_atan (
            gr_complex prompt_s1,
            gr_complex prompt_s2,
            double t1,
            double t2)
```

brief FLL four quadrant arctan discriminator
FLL four quadrant arctan discriminator:

$$\frac{\phi_2 - \phi_1}{t_2 - t1} = \frac{ATAN2(cross, dot)}{t_1 - t_2}, \tag{10.8}$$

where $cross = I_{PS1}Q_{PS2} - I_{PS2}Q_{PS1}$ and $dot = I_{PS1}I_{PS2} + Q_{PS1}Q_{PS2}$, I_{PS1}, Q_{PS1} are the inphase and quadrature prompt correlator outputs respectively at sample time t_1, and I_{PS2}, Q_{PS2} are the inphase and quadrature prompt correlator outputs respectively at sample time t_2. The output is in [radians/second].
References fll_four_quadrant_atan().
Referenced by fll_four_quadrant_atan().

10.13.4.4.10 GetYIntercept()

```
template<typename Fun>
double GetYIntercept (
            Fun && f,
            double x)
```

Definition at line 128 of file tracking_discriminators.h.

10.13.4.4.11 GetYInterceptAbs()

```
template<typename Fun>
double GetYInterceptAbs (
            Fun && f,
            double x)
```

Definition at line 137 of file tracking_discriminators.h.

10.13.4.4.12 phase_unwrap()

```
double phase_unwrap (
            double phase_rad)
```

Phase unwrapping function, input is [rad].
References phase_unwrap().
Referenced by phase_unwrap().

10.13.4.4.13 pll_cloop_two_quadrant_atan()

```
double pll_cloop_two_quadrant_atan (
            gr_complex prompt_s1)
```

PLL Costas loop two quadrant arctan discriminator.
PLL Costas loop two quadrant arctan discriminator:

$$\phi = ATAN\left(\frac{Q_{PS}}{I_{PS}}\right), \tag{10.9}$$

where I_{PS1}, Q_{PS1} are the inphase and quadrature prompt correlator outputs respectively. The output is in [radians].
References pll_cloop_two_quadrant_atan().
Referenced by pll_cloop_two_quadrant_atan().

10.13.4.4.14 pll_four_quadrant_atan()

```
double pll_four_quadrant_atan (
            gr_complex prompt_s1)
```

PLL four quadrant arctan discriminator.
PLL four quadrant arctan discriminator:

$$\phi = ATAN2(Q_{PS}, I_{PS}), \tag{10.10}$$

where I_{PS1}, Q_{PS1} are the inphase and quadrature prompt correlator outputs respectively. The output is in [radians].
References pll_four_quadrant_atan().
Referenced by pll_four_quadrant_atan().

10.13.4.4.15 SinBocCorrelationFunction()

```
template<int M = 1, int N = M>
double SinBocCorrelationFunction (
            double offset_in_chips)
```

Definition at line 150 of file tracking_discriminators.h.

Chapter 11

Namespace Documentation

11.1 block_factory Namespace Reference

Class that produces all kinds of GNSS blocks.

Functions

- std::unique_ptr< SignalSourceInterface > **GetSignalSource** (const ConfigurationInterface *configuration, Concurrent_Queue< pmt::pmt_t > *queue, int ID=-1)
- std::unique_ptr< GNSSBlockInterface > **GetSignalConditioner** (const ConfigurationInterface *configuration, int ID=-1)
- std::vector< std::unique_ptr< GNSSBlockInterface > > **GetChannels** (const ConfigurationInterface *configuration, Concurrent_Queue< pmt::pmt_t > *queue)
- std::unique_ptr< GNSSBlockInterface > **GetObservables** (const ConfigurationInterface *configuration)
- std::unique_ptr< GNSSBlockInterface > **GetPVT** (const ConfigurationInterface *configuration)
- std::unique_ptr< GNSSBlockInterface > GetBlock (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue=nullptr)

  Returns the block with the required role implementation and its configuration parameters.
- std::string get_role_name (const ConfigurationInterface *configuration, const std::string &role_prefix, const std::string &signal, int channel)

  Returns the role name of a channel's block: role_prefix + signal + channel (e.g. "Acquisition_5X12") if that per-channel role has an .implementation property in the configuration, role_prefix + signal (e.g. "Acquisition_5X") otherwise. Use it to read a property of an already built channel block from the same role the block was instantiated with.
- std::unique_ptr< GNSSBlockInterface > **GetChannel** (const ConfigurationInterface *configuration, const std::string &signal, int channel, Concurrent_Queue< pmt::pmt_t > *queue)
- std::unique_ptr< AcquisitionInterface > **GetAcqBlock** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::unique_ptr< TrackingInterface > **GetTrkBlock** (const ConfigurationInterface *configuration, const std::↩ string &role, unsigned int in_streams, unsigned int out_streams)
- std::unique_ptr< TelemetryDecoderInterface > **GetTlmBlock** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)

11.1.1 Detailed Description

Class that produces all kinds of GNSS blocks.

11.1.2 Function Documentation

11.1.2.1 get_role_name()

```
std::string block_factory::get_role_name (
            const ConfigurationInterface * configuration,
            const std::string & role_prefix,
```

```
            const std::string & signal,
            int channel)
```

Returns the role name of a channel's block: role_prefix + signal + channel (e.g. "Acquisition_5X12") if that per-channel role has an .implementation property in the configuration, role_prefix + signal (e.g. "Acquisition_5X") otherwise. Use it to read a property of an already built channel block from the same role the block was instantiated with.
References get_role_name().
Referenced by get_role_name().

11.1.2.2 GetBlock()

```
std::unique_ptr< GNSSBlockInterface > block_factory::GetBlock (
            const ConfigurationInterface * configuration,
            const std::string & role,
            unsigned int in_streams,
            unsigned int out_streams,
            Concurrent_Queue< pmt::pmt_t > * queue = nullptr)
```

Returns the block with the required role implementation and its configuration parameters.
References GetBlock().
Referenced by GetBlock().

11.2 gnss_tow Namespace Reference

Functions

- uint32_t wrap_ms (int64_t tow_ms)
- double wrap_s (double tow_s)
- uint32_t wrap_s_to_ms (double tow_s)
- int64_t to_ms (double tow_s)
- uint32_t seconds_to_ms (int32_t tow_s)
- uint32_t add_ms (uint32_t tow_ms, int64_t delta_ms)
- uint32_t floor_to_second_ms (uint32_t tow_ms)
- uint32_t circular_error_ms (uint32_t lhs_ms, uint32_t rhs_ms)
- int64_t floor_div (int64_t numerator, int64_t denominator)
- bool week_after_delta (uint32_t week, uint32_t tow_ms, int64_t delta_ms, uint32_t &result_week)
- int64_t sample_counter_delta (uint64_t current, uint64_t reference)

Variables

- constexpr uint32_t WEEK_MS = 604800000U
- constexpr double WEEK_S = 604800.0

11.2.1 Detailed Description

Helpers for arithmetic on the GPS time of week reported by the telemetry decoders, which spans [0, 604800) seconds and rolls over at the week boundary.

11.2.2 Function Documentation

11.2.2.1 add_ms()

```
uint32_t gnss_tow::add_ms (
            uint32_t tow_ms,
            int64_t delta_ms)  [inline]
```

Definition at line 75 of file tow_utils.h.

11.2.2.2 circular_error_ms()

```
uint32_t gnss_tow::circular_error_ms (
            uint32_t lhs_ms,
            uint32_t rhs_ms)  [inline]
```

Distance in ms between two TOW values along the shortest direction of the week circle.
Definition at line 91 of file tow_utils.h.
References circular_error_ms().
Referenced by circular_error_ms().

11.2.2.3 floor_div()

```
int64_t gnss_tow::floor_div (
            int64_t numerator,
            int64_t denominator)  [inline]
```

Definition at line 99 of file tow_utils.h.

11.2.2.4 floor_to_second_ms()

```
uint32_t gnss_tow::floor_to_second_ms (
            uint32_t tow_ms)  [inline]
```

Definition at line 81 of file tow_utils.h.

11.2.2.5 sample_counter_delta()

```
int64_t gnss_tow::sample_counter_delta (
            uint64_t current,
            uint64_t reference)  [inline]
```

Definition at line 132 of file tow_utils.h.

11.2.2.6 seconds_to_ms()

```
uint32_t gnss_tow::seconds_to_ms (
            int32_t tow_s)  [inline]
```

Definition at line 69 of file tow_utils.h.

11.2.2.7 to_ms()

```
int64_t gnss_tow::to_ms (
            double tow_s)  [inline]
```

Definition at line 63 of file tow_utils.h.

11.2.2.8 week_after_delta()

```
bool gnss_tow::week_after_delta (
            uint32_t week,
            uint32_t tow_ms,
            int64_t delta_ms,
            uint32_t & result_week)  [inline]
```

Definition at line 111 of file tow_utils.h.

11.2.2.9 wrap_ms()

```
uint32_t gnss_tow::wrap_ms (
            int64_t tow_ms)  [inline]
```

Definition at line 35 of file tow_utils.h.

11.2.2.10 wrap_s()

```
double gnss_tow::wrap_s (
            double tow_s)  [inline]
```

Definition at line 46 of file tow_utils.h.

11.2.2.11 wrap_s_to_ms()

```
uint32_t gnss_tow::wrap_s_to_ms (
            double tow_s)  [inline]
```

Definition at line 57 of file tow_utils.h.

11.2.3 Variable Documentation

11.2.3.1 WEEK_MS

```
uint32_t gnss_tow::WEEK_MS = 604800000U  [constexpr]
```

Definition at line 31 of file tow_utils.h.

11.2.3.2 WEEK_S

```
double gnss_tow::WEEK_S = 604800.0  [constexpr]
```

Definition at line 32 of file tow_utils.h.

11.3 std Namespace Reference

STL namespace.

Classes

- class **allocator**
  STL class.
- class **array**
  STL class.
- class **atomic**
  STL class.
- class **atomic_ref**
  STL class.
- class **auto_ptr**
  STL class.
- class **bad_alloc**
  STL class.
- class **bad_cast**
  STL class.
- class **bad_exception**
  STL class.
- class **bad_typeid**
  STL class.
- class **basic_fstream**
  STL class.
- class **basic_ifstream**
  STL class.
- class **basic_ios**
  STL class.
- class **basic_iostream**

STL class.

- class **basic_istream**

  STL class.
- class **basic_istringstream**

  STL class.
- class **basic_ofstream**

  STL class.
- class **basic_ostream**

  STL class.
- class **basic_ostringstream**

  STL class.
- class **basic_string**

  STL class.
- class **basic_string_view**

  STL class.
- class **basic_stringstream**

  STL class.
- class **bitset**

  STL class.
- class **complex**

  STL class.
- class **deque**

  STL class.
- class **domain_error**

  STL class.
- class **error_category**

  STL class.
- class **error_code**

  STL class.
- class **error_condition**

  STL class.
- class **exception**

  STL class.
- class **forward_list**

  STL class.
- class **fstream**

  STL class.
- struct hash< SensorIdentifier::value_type >
- class **ifstream**

  STL class.
- class **invalid_argument**

  STL class.
- class **ios**

  STL class.
- class **ios_base**

  STL class.
- class **istream**

  STL class.
- class **istringstream**

  STL class.
- class **jthread**

STL class.

- class **length_error**

  STL class.
- class **list**

  STL class.
- class **lock_guard**

  STL class.
- class **logic_error**

  STL class.
- class **map**

  STL class.
- class **multimap**

  STL class.
- class **multiset**

  STL class.
- class **mutex**

  STL class.
- class **ofstream**

  STL class.
- class **ostream**

  STL class.
- class **ostringstream**

  STL class.
- class **out_of_range**

  STL class.
- class **overflow_error**

  STL class.
- class **pair**

  STL class.
- class **priority_queue**

  STL class.
- class **queue**

  STL class.
- class **range_error**

  STL class.
- class **recursive_mutex**

  STL class.
- class **recursive_timed_mutex**

  STL class.
- class **runtime_error**

  STL class.
- class **set**

  STL class.
- class **shared_lock**

  STL class.
- class **shared_mutex**

  STL class.
- class **shared_ptr**

  STL class.
- class **shared_timed_mutex**

  STL class.

- class **smart_ptr**
  STL class.
- class **span**
  STL class.
- class **stack**
  STL class.
- class **string**
  STL class.
- class **string_view**
  STL class.
- class **stringstream**
  STL class.
- class **system_error**
  STL class.
- class **thread**
  STL class.
- class **timed_mutex**
  STL class.
- class **u16string**
  STL class.
- class **u16string_view**
  STL class.
- class **u32string**
  STL class.
- class **u32string_view**
  STL class.
- class **u8string**
  STL class.
- class **u8string_view**
  STL class.
- class **underflow_error**
  STL class.
- class **unique_lock**
  STL class.
- class **unique_ptr**
  STL class.
- class **unordered_map**
  STL class.
- class **unordered_multimap**
  STL class.
- class **unordered_multiset**
  STL class.
- class **unordered_set**
  STL class.
- class **valarray**
  STL class.
- class **vector**
  STL class.
- class **weak_ptr**
  STL class.
- class **wfstream**

STL class.

- class **wifstream**

  STL class.
- class **wios**

  STL class.
- class **wistream**

  STL class.
- class **wistringstream**

  STL class.
- class **wofstream**

  STL class.
- class **wostream**

  STL class.
- class **wostringstream**

  STL class.
- class **wstring**

  STL class.
- class **wstring_view**

  STL class.
- class **wstringstream**

  STL class.

11.3.1 Detailed Description

STL namespace.

Chapter 12

Class Documentation

12.1 sdr_gnss_packet_t::[struct].sats Struct Reference

Public Attributes

- uint8_t **prn**
- uint8_t **snr**
- int16_t **doppler**

12.1.1 Detailed Description

Definition at line 50 of file an_packet_printer.h.
The documentation for this struct was generated from the following files:

12.2 Acq_Conf Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role, double chip_rate, double opt_freq)
- uint32_t GetSamplesPerDwell () const

  *Number of samples processed per dwell: floor(sampled_ms * resampled_fs / 1000), with the window doubled before flooring if bit_transition_flag is set. Computed with integer arithmetic, so it is exact for any resampled_fs.*
- double GetDwellResidualSamples () const

  Fraction of a sample per dwell dropped by GetSamplesPerDwell(), in [0, 1).

Public Attributes

- std::string item_type {"gr_complex"}
- std::string dump_filename
- int64_t fs_in {4000000LL}
- int64_t resampled_fs {0LL}
- size_t it_size {sizeof(gr_complex)}
- float samples_per_ms {0.0}
- float doppler_step2 {125.0}
- float threshold {0.0}
- float pfa {0.0}
- float pfa2 {0.0}
- float samples_per_code {0.0}
- float resampler_ratio {1.0}
- uint32_t sampled_ms {1U}
- uint32_t ms_per_code {1U}
- uint32_t samples_per_chip {2U}

- uint32_t chips_per_second {1023000U}
- uint32_t max_dwells {1U}
- uint32_t num_doppler_bins_step2 {4U}
- uint32_t resampler_latency_samples {0U}
- uint32_t dump_channel {0U}
- int32_t doppler_max {5000}
- int32_t doppler_min {-5000}
- int32_t doppler_step {500}
- bool bit_transition_flag {false}
- bool use_CFAR_algorithm_flag {true}
- bool dump {false}
- bool blocking {true}
- bool blocking_on_standby {false}
- bool make_2_steps {false}
- bool use_automatic_resampler {false}
- bool enable_monitor_output {false}
- uint32_t reference_bin_min_sidelobes {4U}
- bool full_grid_search {false}
- bool use_cuda {false}
- int32_t cuda_device {-1}
- bool acquire_pilot {false}
- bool acquire_iq {false}
- bool cboc {false}
- bool qmboc {false}
- int zero_padding {0}
- uint32_t folding_factor {0}
- uint32_t num_codes {0}
- uint32_t code_length {0}
- uint32_t vector_length {0}

12.2.1 Detailed Description

Definition at line 33 of file acq_conf.h.

12.2.2 Member Function Documentation

12.2.2.1 GetDwellResidualSamples()

```
double Acq_Conf::GetDwellResidualSamples () const
```

Fraction of a sample per dwell dropped by GetSamplesPerDwell(), in [0, 1).

12.2.2.2 GetSamplesPerDwell()

```
uint32_t Acq_Conf::GetSamplesPerDwell () const
```

Number of samples processed per dwell: floor(sampled_ms ∗ resampled_fs / 1000), with the window doubled before flooring if bit_transition_flag is set. Computed with integer arithmetic, so it is exact for any resampled_fs.

12.2.3 Member Data Documentation

12.2.3.1 acquire_iq

```
bool Acq_Conf::acquire_iq {false}
```

Definition at line 104 of file acq_conf.h.

12.2.3.2 acquire_pilot

```
bool Acq_Conf::acquire_pilot {false}
```

Definition at line 103 of file acq_conf.h.

12.2.3.3 bit_transition_flag

```
bool Acq_Conf::bit_transition_flag {false}
```
Definition at line 81 of file acq_conf.h.

12.2.3.4 blocking

```
bool Acq_Conf::blocking {true}
```
Definition at line 84 of file acq_conf.h.

12.2.3.5 blocking_on_standby

```
bool Acq_Conf::blocking_on_standby {false}
```
Definition at line 85 of file acq_conf.h.

12.2.3.6 cboc

```
bool Acq_Conf::cboc {false}
```
Definition at line 105 of file acq_conf.h.

12.2.3.7 chips_per_second

```
uint32_t Acq_Conf::chips_per_second {1023000U}
```
Definition at line 72 of file acq_conf.h.

12.2.3.8 code_length

```
uint32_t Acq_Conf::code_length {0}
```
Definition at line 112 of file acq_conf.h.

12.2.3.9 cuda_device

```
int32_t Acq_Conf::cuda_device {-1}
```
Definition at line 100 of file acq_conf.h.

12.2.3.10 doppler_max

```
int32_t Acq_Conf::doppler_max {5000}
```
Definition at line 77 of file acq_conf.h.

12.2.3.11 doppler_min

```
int32_t Acq_Conf::doppler_min {-5000}
```
Definition at line 78 of file acq_conf.h.

12.2.3.12 doppler_step

```
int32_t Acq_Conf::doppler_step {500}
```
Definition at line 79 of file acq_conf.h.

12.2.3.13 doppler_step2

```
float Acq_Conf::doppler_step2 {125.0}
```
Definition at line 62 of file acq_conf.h.

12.2.3.14 dump

```
bool Acq_Conf::dump {false}
```
Definition at line 83 of file acq_conf.h.

12.2.3.15 dump_channel

```
uint32_t Acq_Conf::dump_channel {0U}
```
Definition at line 76 of file acq_conf.h.

12.2.3.16 dump_filename

```
std::string Acq_Conf::dump_filename
```
Definition at line 54 of file acq_conf.h.

12.2.3.17 enable_monitor_output

```
bool Acq_Conf::enable_monitor_output {false}
```
Definition at line 88 of file acq_conf.h.

12.2.3.18 folding_factor

```
uint32_t Acq_Conf::folding_factor {0}
```
Definition at line 108 of file acq_conf.h.

12.2.3.19 fs_in

```
int64_t Acq_Conf::fs_in {4000000LL}
```
Definition at line 56 of file acq_conf.h.

12.2.3.20 full_grid_search

```
bool Acq_Conf::full_grid_search {false}
```
Definition at line 96 of file acq_conf.h.

12.2.3.21 it_size

```
size_t Acq_Conf::it_size {sizeof(gr_complex)}
```
Definition at line 59 of file acq_conf.h.

12.2.3.22 item_type

```
std::string Acq_Conf::item_type {"gr_complex"}
```
Definition at line 53 of file acq_conf.h.

12.2.3.23 make_2_steps

```
bool Acq_Conf::make_2_steps {false}
```
Definition at line 86 of file acq_conf.h.

12.2.3.24 max_dwells

```
uint32_t Acq_Conf::max_dwells {1U}
```
Definition at line 73 of file acq_conf.h.

12.2.3.25 ms_per_code

```
uint32_t Acq_Conf::ms_per_code {1U}
```
Definition at line 70 of file acq_conf.h.

12.2.3.26 num_codes

```
uint32_t Acq_Conf::num_codes {0}
```
Definition at line 111 of file acq_conf.h.

12.2.3.27 num_doppler_bins_step2

```
uint32_t Acq_Conf::num_doppler_bins_step2 {4U}
```
Definition at line 74 of file acq_conf.h.

12.2.3.28 pfa

```
float Acq_Conf::pfa {0.0}
```
Definition at line 64 of file acq_conf.h.

12.2.3.29 pfa2

```
float Acq_Conf::pfa2 {0.0}
```
Definition at line 65 of file acq_conf.h.

12.2.3.30 qmboc

```
bool Acq_Conf::qmboc {false}
```
Definition at line 106 of file acq_conf.h.

12.2.3.31 reference_bin_min_sidelobes

```
uint32_t Acq_Conf::reference_bin_min_sidelobes {4U}
```
Definition at line 92 of file acq_conf.h.

12.2.3.32 resampled_fs

```
int64_t Acq_Conf::resampled_fs {0LL}
```
Definition at line 57 of file acq_conf.h.

12.2.3.33 resampler_latency_samples

```
uint32_t Acq_Conf::resampler_latency_samples {0U}
```
Definition at line 75 of file acq_conf.h.

12.2.3.34 resampler_ratio

```
float Acq_Conf::resampler_ratio {1.0}
```
Definition at line 67 of file acq_conf.h.

12.2.3.35 sampled_ms

```
uint32_t Acq_Conf::sampled_ms {1U}
```
Definition at line 69 of file acq_conf.h.

12.2.3.36 samples_per_chip

```
uint32_t Acq_Conf::samples_per_chip {2U}
```
Definition at line 71 of file acq_conf.h.

12.2.3.37 samples_per_code

```
float Acq_Conf::samples_per_code {0.0}
```
Definition at line 66 of file acq_conf.h.

12.2.3.38 samples_per_ms

```
float Acq_Conf::samples_per_ms {0.0}
```
Definition at line 61 of file acq_conf.h.

12.2.3.39 threshold

```
float Acq_Conf::threshold {0.0}
```
Definition at line 63 of file acq_conf.h.

12.2.3.40 use_automatic_resampler

```
bool Acq_Conf::use_automatic_resampler {false}
```
Definition at line 87 of file acq_conf.h.

12.2.3.41 use_CFAR_algorithm_flag

```
bool Acq_Conf::use_CFAR_algorithm_flag {true}
```
Definition at line 82 of file acq_conf.h.

12.2.3.42 use_cuda

```
bool Acq_Conf::use_cuda {false}
```
Definition at line 99 of file acq_conf.h.

12.2.3.43 vector_length

```
uint32_t Acq_Conf::vector_length {0}
```
Definition at line 113 of file acq_conf.h.

12.2.3.44 zero_padding

```
int Acq_Conf::zero_padding {0}
```
Definition at line 107 of file acq_conf.h.
The documentation for this class was generated from the following file:

- acq_conf.h

12.3 Acq_Conf_Fpga Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role, uint32_t blk_exp, double code_chips_per_sec, double num_chips_per_code)
- bool **ConfigureAutomaticResampler** (std::vector< std::pair< uint32_t, uint32_t > > downsampling_filter_specs, uint32_t max_FFT_size, double opt_freq)
- bool **Is_acq_config_valid** (uint32_t max_FFT_size) const

Public Attributes

- std::string device_name = "uio0"
- double code_rate_cps
- double code_length_chips
- float doppler_step {250.0}
- float doppler_step2 {125.0}
- float threshold {0.0}
- int64_t fs_in {4000000LL}
- int64_t resampled_fs {4000000LL}
- uint32_t * all_fft_codes = nullptr
- uint32_t num_doppler_bins_step2 {4U}
- uint32_t downsampling_filter_num {0U}
- uint32_t downsampling_factor {1U}
- uint32_t downsampling_filter_delay {0U}
- uint32_t total_block_exp {13U}

- uint32_t excludelimit {5U}
- uint32_t max_num_acqs {2U}
- uint32_t fft_size {1U}
- uint32_t code_length {16000U}
- int32_t doppler_max {5000}
- bool make_2_steps {false}
- bool enable_zero_padding {false}
- bool repeat_satellite {false}

12.3.1 Detailed Description

Definition at line 35 of file acq_conf_fpga.h.

12.3.2 Member Data Documentation

12.3.2.1 all_fft_codes

```
uint32_t* Acq_Conf_Fpga::all_fft_codes = nullptr
```

Definition at line 57 of file acq_conf_fpga.h.

12.3.2.2 code_length

```
uint32_t Acq_Conf_Fpga::code_length {16000U}
```

Definition at line 66 of file acq_conf_fpga.h.

12.3.2.3 code_length_chips

```
double Acq_Conf_Fpga::code_length_chips
```

Definition at line 49 of file acq_conf_fpga.h.

12.3.2.4 code_rate_cps

```
double Acq_Conf_Fpga::code_rate_cps
```

Definition at line 48 of file acq_conf_fpga.h.

12.3.2.5 device_name

```
std::string Acq_Conf_Fpga::device_name = "uio0"
```

Definition at line 47 of file acq_conf_fpga.h.

12.3.2.6 doppler_max

```
int32_t Acq_Conf_Fpga::doppler_max {5000}
```

Definition at line 67 of file acq_conf_fpga.h.

12.3.2.7 doppler_step

```
float Acq_Conf_Fpga::doppler_step {250.0}
```

Definition at line 50 of file acq_conf_fpga.h.

12.3.2.8 doppler_step2

```
float Acq_Conf_Fpga::doppler_step2 {125.0}
```

Definition at line 51 of file acq_conf_fpga.h.

12.3.2.9 downsampling_factor

```
uint32_t Acq_Conf_Fpga::downsampling_factor {1U}
```

Definition at line 60 of file acq_conf_fpga.h.

12.3.2.10 downsampling_filter_delay

```
uint32_t Acq_Conf_Fpga::downsampling_filter_delay {0U}
```
Definition at line 61 of file acq_conf_fpga.h.

12.3.2.11 downsampling_filter_num

```
uint32_t Acq_Conf_Fpga::downsampling_filter_num {0U}
```
Definition at line 59 of file acq_conf_fpga.h.

12.3.2.12 enable_zero_padding

```
bool Acq_Conf_Fpga::enable_zero_padding {false}
```
Definition at line 70 of file acq_conf_fpga.h.

12.3.2.13 excludelimit

```
uint32_t Acq_Conf_Fpga::excludelimit {5U}
```
Definition at line 63 of file acq_conf_fpga.h.

12.3.2.14 fft_size

```
uint32_t Acq_Conf_Fpga::fft_size {1U}
```
Definition at line 65 of file acq_conf_fpga.h.

12.3.2.15 fs_in

```
int64_t Acq_Conf_Fpga::fs_in {4000000LL}
```
Definition at line 54 of file acq_conf_fpga.h.

12.3.2.16 make_2_steps

```
bool Acq_Conf_Fpga::make_2_steps {false}
```
Definition at line 69 of file acq_conf_fpga.h.

12.3.2.17 max_num_acqs

```
uint32_t Acq_Conf_Fpga::max_num_acqs {2U}
```
Definition at line 64 of file acq_conf_fpga.h.

12.3.2.18 num_doppler_bins_step2

```
uint32_t Acq_Conf_Fpga::num_doppler_bins_step2 {4U}
```
Definition at line 58 of file acq_conf_fpga.h.

12.3.2.19 repeat_satellite

```
bool Acq_Conf_Fpga::repeat_satellite {false}
```
Definition at line 71 of file acq_conf_fpga.h.

12.3.2.20 resampled_fs

```
int64_t Acq_Conf_Fpga::resampled_fs {4000000LL}
```
Definition at line 55 of file acq_conf_fpga.h.

12.3.2.21 threshold

```
float Acq_Conf_Fpga::threshold {0.0}
```
Definition at line 52 of file acq_conf_fpga.h.

12.3.2.22 total_block_exp

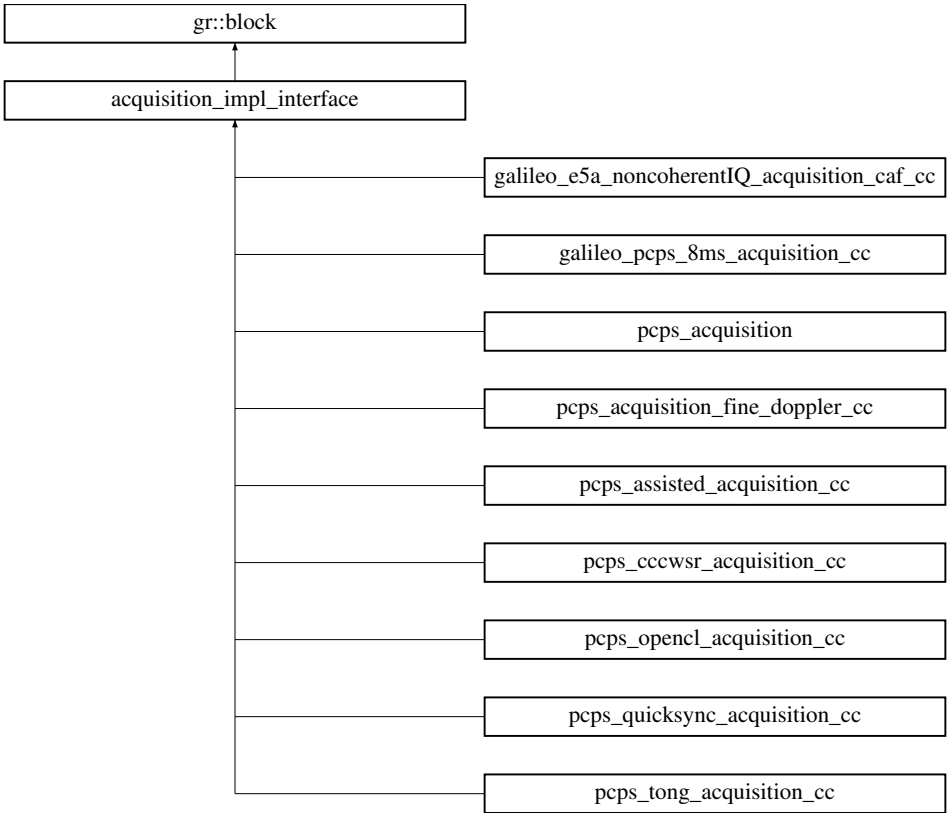
```
uint32_t Acq_Conf_Fpga::total_block_exp {13U}
```

Definition at line 62 of file acq_conf_fpga.h.
The documentation for this class was generated from the following file:

- acq_conf_fpga.h

12.4 acquisition_impl_interface Class Reference

This abstract class represents an interface to an acquisition GNSS block.
`#include <acquisition_impl_interface.h>`
Inheritance diagram for acquisition_impl_interface:



Public Member Functions

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_↩ signature::sptr output_signature)
- virtual void set_gnss_synchro (Gnss_Synchro ∗gnss_synchro)=0
- virtual void set_channel (uint32_t channel_id)=0
- virtual void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm)=0
- virtual void set_local_code (std::complex< float > ∗)
- virtual void set_local_code (std::complex< float > ∗, std::complex< float > ∗)
- virtual uint32_t mag () const =0
- virtual void set_active (bool active)=0

12.4.1 Detailed Description

This abstract class represents an interface to an acquisition GNSS block.

Abstract class for acquisition algorithms. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 50 of file acquisition_impl_interface.h.

12.4.2 Constructor & Destructor Documentation

12.4.2.1 acquisition_impl_interface()

```
acquisition_impl_interface::acquisition_impl_interface (
            const std::string & name,
            gr::io_signature::sptr input_signature,
            gr::io_signature::sptr output_signature)  [inline]
```

Definition at line 53 of file acquisition_impl_interface.h.

12.4.3 Member Function Documentation

12.4.3.1 mag()

```
virtual uint32_t acquisition_impl_interface::mag () const  [pure virtual]
```

Implemented in galileo_e5a_noncoherentIQ_acquisition_caf_cc, galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_acquisition_fine_doppler_cc, pcps_assisted_acquisition_cc, pcps_cccwsr_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.

12.4.3.2 set_active()

```
virtual void acquisition_impl_interface::set_active (
            bool active)  [pure virtual]
```

Implemented in galileo_e5a_noncoherentIQ_acquisition_caf_cc, galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_acquisition_fine_doppler_cc, pcps_assisted_acquisition_cc, pcps_cccwsr_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.

12.4.3.3 set_channel()

```
virtual void acquisition_impl_interface::set_channel (
            uint32_t channel_id)  [pure virtual]
```

Implemented in galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_assisted_acquisition_cc, pcps_cccwsr_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.

12.4.3.4 set_channel_fsm()

```
virtual void acquisition_impl_interface::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [pure virtual]
```

Implemented in galileo_e5a_noncoherentIQ_acquisition_caf_cc, galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_acquisition_fine_doppler_cc, pcps_assisted_acquisition_cc, pcps_cccwsr_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.

12.4.3.5 set_gnss_synchro()

```
virtual void acquisition_impl_interface::set_gnss_synchro (
            Gnss_Synchro * gnss_synchro)  [pure virtual]
```

Implemented in galileo_e5a_noncoherentIQ_acquisition_caf_cc, galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_acquisition_fine_doppler_cc, pcps_assisted_acquisition_cc, pcps_cccwsr_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.

12.4.3.6 set_local_code() [1/2]

```
virtual void acquisition_impl_interface::set_local_code (
            std::complex< float > * )  [inline], [virtual]
```

Reimplemented in galileo_pcps_8ms_acquisition_cc, pcps_acquisition, pcps_acquisition_fine_doppler_cc, pcps_assisted_acquisition_cc, pcps_opencl_acquisition_cc, pcps_quicksync_acquisition_cc, and pcps_tong_acquisition_cc.
Definition at line 60 of file acquisition_impl_interface.h.

12.4.3.7 set_local_code() [2/2]

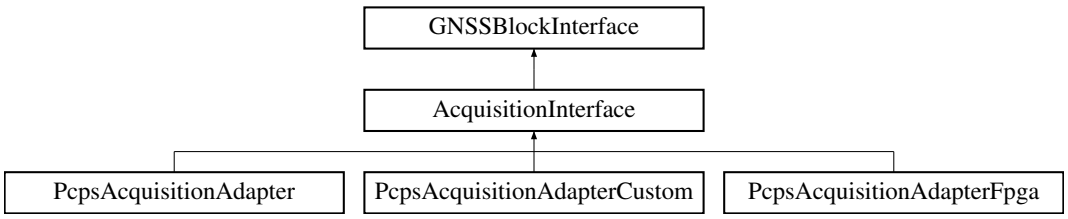
```
virtual void acquisition_impl_interface::set_local_code (
            std::complex< float > * ,
            std::complex< float > * )  [inline], [virtual]
```

Reimplemented in galileo_e5a_noncoherentIQ_acquisition_caf_cc, and pcps_cccwsr_acquisition_cc.
Definition at line 61 of file acquisition_impl_interface.h.
The documentation for this class was generated from the following file:

- acquisition_impl_interface.h

12.5 AcquisitionInterface Class Reference

This abstract class represents an interface to an acquisition GNSS block.
`#include <acquisition_interface.h>`
Inheritance diagram for AcquisitionInterface:



Public Member Functions

- virtual void set_gnss_synchro (Gnss_Synchro *gnss_synchro)=0
- virtual void set_channel (unsigned int channel_id)=0
- virtual void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm)=0
- virtual void set_doppler_center (int)
- virtual void set_doppler_num_bins (unsigned int)

  Number of Doppler bins to search, centered on set_doppler_center(). doppler_num_bins == 0 searches the full configured Doppler range (computed from doppler_max/doppler_step – the one value a caller can't supply directly, since that count is implementation-private); any other value is the literal candidate bin count to search – 1 for an exactly-known Doppler (as when assisted by an already-tracked primary frequency or a visibility-aware prediction), or any N in between for a search with partial uncertainty.
- virtual void set_local_code ()=0
- virtual signed int mag ()=0
- virtual void reset ()=0
- virtual void stop_acquisition ()=0
- virtual void set_resampler_latency (uint32_t latency_samples)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.5.1 Detailed Description

This abstract class represents an interface to an acquisition GNSS block.
Abstract class for acquisition algorithms. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 50 of file acquisition_interface.h.

12.5.2 Member Function Documentation

12.5.2.1 mag()

```
virtual signed int AcquisitionInterface::mag ()  [pure virtual]
```
Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.2 reset()

```
virtual void AcquisitionInterface::reset ()  [pure virtual]
```
Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.3 set_channel()

```
virtual void AcquisitionInterface::set_channel (
            unsigned int channel_id)  [pure virtual]
```
Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.4 set_channel_fsm()

```
virtual void AcquisitionInterface::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [pure virtual]
```
Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.5 set_doppler_center()

```
virtual void AcquisitionInterface::set_doppler_center (
            int )  [inline], [virtual]
```
Reimplemented in PcpsAcquisitionAdapter.
Definition at line 56 of file acquisition_interface.h.

12.5.2.6 set_doppler_num_bins()

```
virtual void AcquisitionInterface::set_doppler_num_bins (
            unsigned int )  [inline], [virtual]
```
Number of Doppler bins to search, centered on set_doppler_center(). doppler_num_bins == 0 searches the full configured Doppler range (computed from doppler_max/doppler_step – the one value a caller can't supply directly, since that count is implementation-private); any other value is the literal candidate bin count to search – 1 for an exactly-known Doppler (as when assisted by an already-tracked primary frequency or a visibility-aware prediction), or any N in between for a search with partial uncertainty.
Reimplemented in PcpsAcquisitionAdapter.
Definition at line 65 of file acquisition_interface.h.

12.5.2.7 set_gnss_synchro()

```
virtual void AcquisitionInterface::set_gnss_synchro (
            Gnss_Synchro * gnss_synchro)  [pure virtual]
```

Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.8 set_local_code()

```
virtual void AcquisitionInterface::set_local_code ()  [pure virtual]
```

Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.

12.5.2.9 set_resampler_latency()

```
virtual void AcquisitionInterface::set_resampler_latency (
            uint32_t latency_samples)  [pure virtual]
```

Implemented in PcpsAcquisitionAdapter.

12.5.2.10 stop_acquisition()

```
virtual void AcquisitionInterface::stop_acquisition ()  [pure virtual]
```

Implemented in PcpsAcquisitionAdapter, and PcpsAcquisitionAdapterCustom.
The documentation for this class was generated from the following file:

- acquisition_interface.h

12.6 ad936x_iio_custom Class Reference

Public Member Functions

- **ad936x_iio_custom** (int debug_level_, int log_level_)
- bool **initialize_device** (const std::string &pluto_device_uri, const std::string &board_type)
- bool **init_config_ad9361_rx** (long long bandwidth_, long long sample_rate_, long long freq_, const std::string &rf_port_select_, const std::string &rf_filter, const std::string &gain_mode_rx0_, const std::string &gain_↩ mode_rx1_, double rf_gain_rx0_, double rf_gain_rx1_, bool enable_ch0, bool enable_ch1, long long freq_↩ 2ch, double lo_attenuation_db_, bool high_side_lo_, int tx_lo_channel_)
- bool **calibrate** (int ch, double bw_hz)
- double **get_rx_gain** (int ch_num)
- bool **setRXGain** (int ch_num, const std::string &gain_mode, double gain_dB)
- bool **set_antenna_port** (int ch, int antenna_idx)
- double **get_frequency** (int ch)
- bool **set_frequency** (int ch, double freq_hz)
- bool **start_sample_rx** (bool ppsmode)
- void **stop_record** ()
- void **set_gnsstime_queue** (std::shared_ptr< Concurrent_Queue< GnssTime > > queue)
- void **set_pps_samplestamp_queue** (std::shared_ptr< Concurrent_Queue< PpsSamplestamp > > queue)
- bool **get_rx_frequency** (long long &freq_hz)
- bool **set_rx_frequency** (long long freq_hz)
- bool **read_die_temp** (double &temp_c)
- void **pop_sample_buffer** (std::shared_ptr< ad936x_iio_samples > &current_buffer)
- void **push_sample_buffer** (std::shared_ptr< ad936x_iio_samples > &current_buffer)

Public Attributes

- int n_channels

12.6.1 Detailed Description

Definition at line 39 of file ad936x_iio_custom.h.

12.6.2 Member Data Documentation

12.6.2.1 n_channels

```
int ad936x_iio_custom::n_channels
```
Definition at line 84 of file ad936x_iio_custom.h.
The documentation for this class was generated from the following file:

- ad936x_iio_custom.h

12.7 ad936x_iio_samples Class Reference

Public Attributes

- uint32_t n_bytes {0}
- uint32_t n_interleaved_iq_samples {0}
- uint16_t n_channels {0}
- uint16_t step_bytes {0}
- char buffer [IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES ∗4 ∗4]

12.7.1 Detailed Description

Definition at line 33 of file ad936x_iio_samples.h.

12.7.2 Member Data Documentation

12.7.2.1 buffer

```
char ad936x_iio_samples::buffer[IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES *4 *4]
```
Definition at line 41 of file ad936x_iio_samples.h.

12.7.2.2 n_bytes

```
uint32_t ad936x_iio_samples::n_bytes {0}
```
Definition at line 37 of file ad936x_iio_samples.h.

12.7.2.3 n_channels

```
uint16_t ad936x_iio_samples::n_channels {0}
```
Definition at line 39 of file ad936x_iio_samples.h.

12.7.2.4 n_interleaved_iq_samples

```
uint32_t ad936x_iio_samples::n_interleaved_iq_samples {0}
```
Definition at line 38 of file ad936x_iio_samples.h.

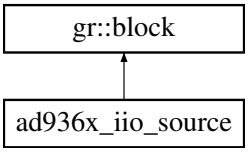
12.7.2.5 step_bytes

```
uint16_t ad936x_iio_samples::step_bytes {0}
```
Definition at line 40 of file ad936x_iio_samples.h.
The documentation for this class was generated from the following file:

- ad936x_iio_samples.h

12.8 ad936x_iio_source Class Reference

This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.
`#include <ad936x_iio_source.h>`
Inheritance diagram for ad936x_iio_source:



Public Member Functions

- bool start ()

  start the sample transmission
- bool stop ()

  stop the sample transmission
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- ad936x_iio_source_sptr **ad936x_iio_make_source_sptr** (const std::string &pluto_uri_, const std::string &board_type_, int64_t bandwidth_, int64_t sample_rate_, int64_t freq_, const std::string &rf_port_select_, const std::string &rf_filter, const std::string &gain_mode_rx0_, const std::string &gain_mode_rx1_, double rf↩ _gain_rx0_, double rf_gain_rx1_, bool enable_ch0, bool enable_ch1, int64_t freq_2ch, bool ppsmode_, bool customsamplesize_, const std::string &fe_ip_, int fe_ctlport_, int ssize_, int bshift_, bool spattern_, double lo_attenuation_db_, bool high_side_lo_, int tx_lo_channel_)

12.8.1 Detailed Description

This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.
Definition at line 77 of file ad936x_iio_source.h.

12.8.2 Member Function Documentation

12.8.2.1 start()

```
bool ad936x_iio_source::start ()
```

start the sample transmission

12.8.2.2 stop()

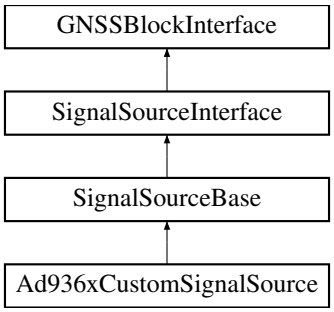
```
bool ad936x_iio_source::stop ()
```

stop the sample transmission
The documentation for this class was generated from the following file:

- ad936x_iio_source.h

12.9 Ad936xCustomSignalSource Class Reference

This class instantiates the Ad936xCustom gnuradio signal source. It has support also for a customized Ad936x↩ Custom firmware and signal source to support PPS samplestamp reading.
`#include <ad936x_custom_signal_source.h>`
Inheritance diagram for Ad936xCustomSignalSource:



Public Member Functions

- **Ad936xCustomSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.9.1 Detailed Description

This class instantiates the Ad936xCustom gnuradio signal source. It has support also for a customized Ad936x↩ Custom firmware and signal source to support PPS samplestamp reading.
Definition at line 51 of file ad936x_custom_signal_source.h.

12.9.2 Member Function Documentation

12.9.2.1 connect()

```
void Ad936xCustomSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.9.2.2 disconnect()

```
void Ad936xCustomSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.9.2.3 get_left_block()

```
gr::basic_block_sptr Ad936xCustomSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.9.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr Ad936xCustomSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.9.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr Ad936xCustomSignalSource::get_right_block (
            int RF_channel)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.9.2.6 item_size()

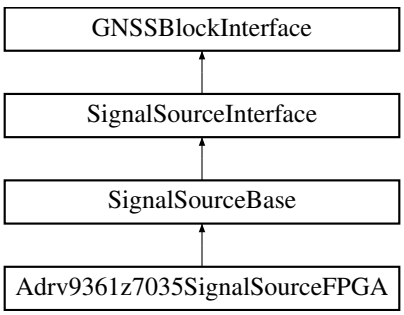
```
size_t Ad936xCustomSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 60 of file ad936x_custom_signal_source.h.
The documentation for this class was generated from the following file:

- ad936x_custom_signal_source.h

12.10 Adrv9361z7035SignalSourceFPGA Class Reference

Inheritance diagram for Adrv9361z7035SignalSourceFPGA:



Public Member Functions

- **Adrv9361z7035SignalSourceFPGA** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.10.1 Detailed Description

Definition at line 46 of file adrv9361_z7035_signal_source_fpga.h.

12.10.2 Member Function Documentation

12.10.2.1 connect()

```
void Adrv9361z7035SignalSourceFPGA::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.10.2.2 disconnect()

```
void Adrv9361z7035SignalSourceFPGA::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.10.2.3 get_left_block()

```
gr::basic_block_sptr Adrv9361z7035SignalSourceFPGA::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.10.2.4 get_right_block()

```
gr::basic_block_sptr Adrv9361z7035SignalSourceFPGA::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.10.2.5 item_size()

```
size_t Adrv9361z7035SignalSourceFPGA::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 55 of file adrv9361_z7035_signal_source_fpga.h.
The documentation for this class was generated from the following file:

- adrv9361_z7035_signal_source_fpga.h

12.11 Agnss_Ref_Location Class Reference

Interface of an Assisted GNSS REFERENCE LOCATION storage.
`#include <agnss_ref_location.h>`

Public Member Functions

- Agnss_Ref_Location ()=default
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the Ref location on disk file.

Public Attributes

- double lat {}
- double lon {}
- double uncertainty {}
- bool valid {}

12.11.1 Detailed Description

Interface of an Assisted GNSS REFERENCE LOCATION storage.
Definition at line 36 of file agnss_ref_location.h.

12.11.2 Constructor & Destructor Documentation

12.11.2.1 Agnss_Ref_Location()

```
Agnss_Ref_Location::Agnss_Ref_Location ()  [default]
```

Default constructor

12.11.3 Member Function Documentation

12.11.3.1 serialize()

```
template<class Archive>
void Agnss_Ref_Location::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the Ref location on disk file.
Definition at line 55 of file agnss_ref_location.h.

12.11.4 Member Data Documentation

12.11.4.1 lat

```
double Agnss_Ref_Location::lat {}
```

Definition at line 44 of file agnss_ref_location.h.

12.11.4.2 lon

```
double Agnss_Ref_Location::lon {}
```

Definition at line 45 of file agnss_ref_location.h.

12.11.4.3 uncertainty

```
double Agnss_Ref_Location::uncertainty {}
```

Definition at line 46 of file agnss_ref_location.h.

12.11.4.4 valid

```
bool Agnss_Ref_Location::valid {}
```
Definition at line 47 of file agnss_ref_location.h.
The documentation for this class was generated from the following file:

- agnss_ref_location.h

12.12 Agnss_Ref_Time Class Reference

Interface of an Assisted GNSS REFERENCE TIME storage.
`#include <agnss_ref_time.h>`

Public Member Functions

- Agnss_Ref_Time ()=default
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ref time data on disk file.

Public Attributes

- double tow {}
- double week {}
- double seconds {}
- double microseconds {}
- bool valid {}

12.12.1 Detailed Description

Interface of an Assisted GNSS REFERENCE TIME storage.
Definition at line 35 of file agnss_ref_time.h.

12.12.2 Constructor & Destructor Documentation

12.12.2.1 Agnss_Ref_Time()

```
Agnss_Ref_Time::Agnss_Ref_Time ()  [default]
```
Default constructor

12.12.3 Member Function Documentation

12.12.3.1 serialize()

```
template<class Archive>
void Agnss_Ref_Time::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```
Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ref time data on disk file.
Definition at line 55 of file agnss_ref_time.h.

12.12.4 Member Data Documentation

12.12.4.1 microseconds

```
double Agnss_Ref_Time::microseconds {}
```
Definition at line 46 of file agnss_ref_time.h.

12.12.4.2 seconds

```
double Agnss_Ref_Time::seconds {}
```
Definition at line 45 of file agnss_ref_time.h.

12.12.4.3 tow

```
double Agnss_Ref_Time::tow {}
```
Definition at line 43 of file agnss_ref_time.h.

12.12.4.4 valid

```
bool Agnss_Ref_Time::valid {}
```
Definition at line 47 of file agnss_ref_time.h.

12.12.4.5 week

```
double Agnss_Ref_Time::week {}
```
Definition at line 44 of file agnss_ref_time.h.
The documentation for this class was generated from the following file:

- agnss_ref_time.h

12.13 Allocator Struct Reference

Public Member Functions

- Allocator (std::size_t countwords, void ∗&buffer)
- template<typename WordType>
  void operator() () const

Public Attributes

- std::size_t countwords_
- void ∗& buffer_

12.13.1 Detailed Description

Definition at line 58 of file ion_gsms_chunk_data.h.

12.13.2 Constructor & Destructor Documentation

12.13.2.1 Allocator()

```
Allocator::Allocator (
            std::size_t countwords,
            void *& buffer)  [inline]
```
Definition at line 63 of file ion_gsms_chunk_data.h.

12.13.3 Member Function Documentation

12.13.3.1 operator()()

```
template<typename WordType>
void Allocator::operator() () const  [inline]
```
Definition at line 67 of file ion_gsms_chunk_data.h.

12.13.4 Member Data Documentation

12.13.4.1 buffer_

`void*& Allocator::buffer_`
Definition at line 61 of file ion_gsms_chunk_data.h.

12.13.4.2 countwords_

`std::size_t Allocator::countwords_`
Definition at line 60 of file ion_gsms_chunk_data.h.
The documentation for this struct was generated from the following file:

- ion_gsms_chunk_data.h

12.14 alm_t Struct Reference

Public Attributes

- int sat
- int svh
- int svconf
- int week
- gtime_t toa
- double A
- double e
- double i0
- double OMG0
- double omg
- double M0
- double OMGd
- double toas
- double f0
- double f1

12.14.1 Detailed Description

Definition at line 447 of file rtklib.h.

12.14.2 Member Data Documentation

12.14.2.1 A

`double alm_t::A`
Definition at line 455 of file rtklib.h.

12.14.2.2 e

`double alm_t::e`
Definition at line 455 of file rtklib.h.

12.14.2.3 f0

`double alm_t::f0`
Definition at line 457 of file rtklib.h.

12.14.2.4 f1

`double alm_t::f1`
Definition at line 457 of file rtklib.h.

12.14.2.5 i0

`double alm_t::i0`
Definition at line 455 of file rtklib.h.

12.14.2.6 M0

`double alm_t::M0`
Definition at line 455 of file rtklib.h.

12.14.2.7 omg

`double alm_t::omg`
Definition at line 455 of file rtklib.h.

12.14.2.8 OMG0

`double alm_t::OMG0`
Definition at line 455 of file rtklib.h.

12.14.2.9 OMGd

`double alm_t::OMGd`
Definition at line 455 of file rtklib.h.

12.14.2.10 sat

`int alm_t::sat`
Definition at line 449 of file rtklib.h.

12.14.2.11 svconf

`int alm_t::svconf`
Definition at line 451 of file rtklib.h.

12.14.2.12 svh

`int alm_t::svh`
Definition at line 450 of file rtklib.h.

12.14.2.13 toa

`gtime_t alm_t::toa`
Definition at line 453 of file rtklib.h.

12.14.2.14 toas

`double alm_t::toas`
Definition at line 456 of file rtklib.h.

12.14.2.15 week

`int alm_t::week`
Definition at line 452 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.15 ambc_t Struct Reference

Public Attributes

- gtime_t epoch [4]
- int n [4]
- double LC [4]
- double LCv [4]
- int fixcnt
- char flags [MAXSAT]

12.15.1 Detailed Description

Definition at line 1118 of file rtklib.h.

12.15.2 Member Data Documentation

12.15.2.1 epoch

```
gtime_t ambc_t::epoch[4]
```

Definition at line 1120 of file rtklib.h.

12.15.2.2 fixcnt

```
int ambc_t::fixcnt
```

Definition at line 1124 of file rtklib.h.

12.15.2.3 flags

```
char ambc_t::flags[MAXSAT]
```

Definition at line 1125 of file rtklib.h.

12.15.2.4 LC

```
double ambc_t::LC[4]
```

Definition at line 1122 of file rtklib.h.

12.15.2.5 LCv

```
double ambc_t::LCv[4]
```

Definition at line 1123 of file rtklib.h.

12.15.2.6 n

```
int ambc_t::n[4]
```

Definition at line 1121 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.16 An_Packet_Printer Class Reference

class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.
`#include <an_packet_printer.h>`

Public Member Functions

- An_Packet_Printer (const std::string &an_dump_devname)

  Default constructor.
- ∼An_Packet_Printer ()

  Default destructor.
- bool print_packet (const Rtklib_Solver *const pvt_data, const std::map< int, Gnss_Synchro > &gnss_↩ observables_map)

  Print AN packet to the initialized device.
- void close_serial () const

  Close serial port. Also done in the destructor, this is only for testing.

12.16.1 Detailed Description

class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.
Definition at line 73 of file an_packet_printer.h.

12.16.2 Constructor & Destructor Documentation

12.16.2.1 An_Packet_Printer()

```
An_Packet_Printer::An_Packet_Printer (
            const std::string & an_dump_devname)  [explicit]
```

Default constructor.

12.16.2.2 ∼An_Packet_Printer()

```
An_Packet_Printer::∼An_Packet_Printer ()
```

Default destructor.

12.16.3 Member Function Documentation

12.16.3.1 close_serial()

```
void An_Packet_Printer::close_serial () const
```

Close serial port. Also done in the destructor, this is only for testing.

12.16.3.2 print_packet()

```
bool An_Packet_Printer::print_packet (
            const Rtklib_Solver *const pvt_data,
            const std::map< int, Gnss_Synchro > & gnss_observables_map)
```

Print AN packet to the initialized device.
The documentation for this class was generated from the following file:

- an_packet_printer.h

12.17 an_packet_t Struct Reference

Public Attributes

- uint8_t header [4]
- uint8_t data [113]

12.17.1 Detailed Description

Definition at line 62 of file an_packet_printer.h.

12.17.2 Member Data Documentation

12.17.2.1 data

`uint8_t an_packet_t::data[113]`
Definition at line 65 of file an_packet_printer.h.

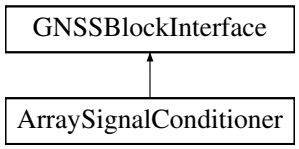
12.17.2.2 header

`uint8_t an_packet_t::header[4]`
Definition at line 64 of file an_packet_printer.h.
The documentation for this struct was generated from the following file:

- an_packet_printer.h

12.18 ArraySignalConditioner Class Reference

This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.
`#include <array_signal_conditioner.h>`
Inheritance diagram for ArraySignalConditioner:



Public Member Functions

- ArraySignalConditioner (std::shared_ptr< GNSSBlockInterface > data_type_adapt, std::shared_ptr< GNSSBlockInterface > in_filt, std::shared_ptr< GNSSBlockInterface > res, std::string role)

  Constructor.
- ~ArraySignalConditioner ()=default

  Destructor.
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string role () override
- std::string implementation () override

  Returns "Array_Signal_Conditioner".
- size_t item_size () override
- std::shared_ptr< GNSSBlockInterface > data_type_adapter ()
- std::shared_ptr< GNSSBlockInterface > input_filter ()
- std::shared_ptr< GNSSBlockInterface > resampler ()

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.18.1 Detailed Description

This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.
Definition at line 41 of file array_signal_conditioner.h.

12.18.2 Constructor & Destructor Documentation

12.18.2.1 ArraySignalConditioner()

```
ArraySignalConditioner::ArraySignalConditioner (
            std::shared_ptr< GNSSBlockInterface > data_type_adapt,
            std::shared_ptr< GNSSBlockInterface > in_filt,
            std::shared_ptr< GNSSBlockInterface > res,
            std::string role)
```

Constructor.

12.18.2.2 ~ArraySignalConditioner()

```
ArraySignalConditioner::~ArraySignalConditioner ()  [default]
```

Destructor.

12.18.3 Member Function Documentation

12.18.3.1 connect()

```
void ArraySignalConditioner::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.18.3.2 data_type_adapter()

```
std::shared_ptr< GNSSBlockInterface > ArraySignalConditioner::data_type_adapter ()  [inline]
```

Definition at line 63 of file array_signal_conditioner.h.

12.18.3.3 disconnect()

```
void ArraySignalConditioner::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.18.3.4 get_left_block()

```
gr::basic_block_sptr ArraySignalConditioner::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.18.3.5 get_right_block()

```
gr::basic_block_sptr ArraySignalConditioner::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.18.3.6 implementation()

```
std::string ArraySignalConditioner::implementation ()  [inline], [override], [virtual]
```

Returns "Array_Signal_Conditioner".
Implements GNSSBlockInterface.
Definition at line 60 of file array_signal_conditioner.h.

12.18.3.7 input_filter()

`std::shared_ptr< GNSSBlockInterface > ArraySignalConditioner::input_filter () [inline]`
Definition at line 64 of file array_signal_conditioner.h.

12.18.3.8 item_size()

`size_t ArraySignalConditioner::item_size () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 61 of file array_signal_conditioner.h.

12.18.3.9 resampler()

`std::shared_ptr< GNSSBlockInterface > ArraySignalConditioner::resampler () [inline]`
Definition at line 65 of file array_signal_conditioner.h.

12.18.3.10 role()

`std::string ArraySignalConditioner::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 58 of file array_signal_conditioner.h.
The documentation for this class was generated from the following file:

- array_signal_conditioner.h

12.19 Bayesian_estimator Class Reference

Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance).
`#include <bayesian_estimation.h>`

Public Member Functions

- **Bayesian_estimator** (int ny)
- **Bayesian_estimator** (const arma::vec &mu_prior_0, int kappa_prior_0, int nu_prior_0, const arma::mat &Psi_prior_0)
- void **init** (const arma::mat &mu_prior_0, int kappa_prior_0, int nu_prior_0, const arma::mat &Psi_prior_0)
- void **update_sequential** (const arma::vec &data)
- void **update_sequential** (const arma::vec &data, const arma::vec &mu_prior_0, int kappa_prior_0, int nu_↩ prior_0, const arma::mat &Psi_prior_0)
- arma::mat **get_mu_est** () const
- arma::mat **get_Psi_est** () const

12.19.1 Detailed Description

Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance).
Bayesian_estimator is an estimator which performs estimation of noise characteristics from a sequence of identically and independently distributed (IID) samples of a stationary stochastic process by way of Bayesian inference using conjugate priors. The posterior distribution is assumed to be Gaussian with mean \mathbf{\mu} and covariance \hat{\mathbf{C}}, which has a conjugate prior given by a normal-inverse-Wishart distribution with parameters \mathbf{\mu}_{0}, \kappa_{0}, \nu_{0}, and \mathbf{\Psi}.
[1] TODO: Ref1
Definition at line 60 of file bayesian_estimation.h.
The documentation for this class was generated from the following file:

- bayesian_estimation.h

12.20 Bds3_B1c_AlmanacMedium Struct Reference

Public Attributes

- int32_t prn {}
- int32_t sat_type {}
- int32_t wna {}
- int32_t toa_s {}
- double eccentricity {}
- double delta_i_rad {}
- double sqrt_a_m_sqrt {}
- double omega0_rad {}
- double omega_dot_rad_s {}
- double omega_rad {}
- double m0_rad {}
- double af0_s {}
- double af1_s_s {}
- int32_t health {}

12.20.1 Detailed Description

Definition at line 54 of file beidou_cnav1_navigation_message.h.

12.20.2 Member Data Documentation

12.20.2.1 af0_s

```
double Bds3_B1c_AlmanacMedium::af0_s {}
```

Definition at line 67 of file beidou_cnav1_navigation_message.h.

12.20.2.2 af1_s_s

```
double Bds3_B1c_AlmanacMedium::af1_s_s {}
```

Definition at line 68 of file beidou_cnav1_navigation_message.h.

12.20.2.3 delta_i_rad

```
double Bds3_B1c_AlmanacMedium::delta_i_rad {}
```

Definition at line 61 of file beidou_cnav1_navigation_message.h.

12.20.2.4 eccentricity

```
double Bds3_B1c_AlmanacMedium::eccentricity {}
```

Definition at line 60 of file beidou_cnav1_navigation_message.h.

12.20.2.5 health

```
int32_t Bds3_B1c_AlmanacMedium::health {}
```

Definition at line 69 of file beidou_cnav1_navigation_message.h.

12.20.2.6 m0_rad

```
double Bds3_B1c_AlmanacMedium::m0_rad {}
```

Definition at line 66 of file beidou_cnav1_navigation_message.h.

12.20.2.7 omega0_rad

```
double Bds3_B1c_AlmanacMedium::omega0_rad {}
```

Definition at line 63 of file beidou_cnav1_navigation_message.h.

12.20.2.8 omega_dot_rad_s

```
double Bds3_B1c_AlmanacMedium::omega_dot_rad_s {}
```
Definition at line 64 of file beidou_cnav1_navigation_message.h.

12.20.2.9 omega_rad

```
double Bds3_B1c_AlmanacMedium::omega_rad {}
```
Definition at line 65 of file beidou_cnav1_navigation_message.h.

12.20.2.10 prn

```
int32_t Bds3_B1c_AlmanacMedium::prn {}
```
Definition at line 56 of file beidou_cnav1_navigation_message.h.

12.20.2.11 sat_type

```
int32_t Bds3_B1c_AlmanacMedium::sat_type {}
```
Definition at line 57 of file beidou_cnav1_navigation_message.h.

12.20.2.12 sqrt_a_m_sqrt

```
double Bds3_B1c_AlmanacMedium::sqrt_a_m_sqrt {}
```
Definition at line 62 of file beidou_cnav1_navigation_message.h.

12.20.2.13 toa_s

```
int32_t Bds3_B1c_AlmanacMedium::toa_s {}
```
Definition at line 59 of file beidou_cnav1_navigation_message.h.

12.20.2.14 wna

```
int32_t Bds3_B1c_AlmanacMedium::wna {}
```
Definition at line 58 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.21 Bds3_B1c_AlmanacReduced Struct Reference

Public Attributes

- int32_t prn {}
- int32_t sat_type {}
- double delta_a_m {}
- double omega0_rad {}
- double phi0_rad {}
- int32_t health {}

12.21.1 Detailed Description

Definition at line 43 of file beidou_cnav1_navigation_message.h.

12.21.2 Member Data Documentation

12.21.2.1 delta_a_m

```
double Bds3_B1c_AlmanacReduced::delta_a_m {}
```
Definition at line 47 of file beidou_cnav1_navigation_message.h.

12.21.2.2 health

```
int32_t Bds3_B1c_AlmanacReduced::health {}
```
Definition at line 50 of file beidou_cnav1_navigation_message.h.

12.21.2.3 omega0_rad

```
double Bds3_B1c_AlmanacReduced::omega0_rad {}
```
Definition at line 48 of file beidou_cnav1_navigation_message.h.

12.21.2.4 phi0_rad

```
double Bds3_B1c_AlmanacReduced::phi0_rad {}
```
Definition at line 49 of file beidou_cnav1_navigation_message.h.

12.21.2.5 prn

```
int32_t Bds3_B1c_AlmanacReduced::prn {}
```
Definition at line 45 of file beidou_cnav1_navigation_message.h.

12.21.2.6 sat_type

```
int32_t Bds3_B1c_AlmanacReduced::sat_type {}
```
Definition at line 46 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.22 Bds3_B1c_Bgto Struct Reference

Public Attributes

- int32_t gnss_id {}
- int32_t wn0_bgto {}
- int32_t t0_bgto_s {}
- double a0_bgto_s {}
- double a1_bgto_s_s {}
- double a2_bgto_s_s2 {}

12.22.1 Detailed Description

Definition at line 85 of file beidou_cnav1_navigation_message.h.

12.22.2 Member Data Documentation

12.22.2.1 a0_bgto_s

```
double Bds3_B1c_Bgto::a0_bgto_s {}
```
Definition at line 90 of file beidou_cnav1_navigation_message.h.

12.22.2.2 a1_bgto_s_s

```
double Bds3_B1c_Bgto::a1_bgto_s_s {}
```
Definition at line 91 of file beidou_cnav1_navigation_message.h.

12.22.2.3 a2_bgto_s_s2

```
double Bds3_B1c_Bgto::a2_bgto_s_s2 {}
```
Definition at line 92 of file beidou_cnav1_navigation_message.h.

12.22.2.4 gnss_id

```
int32_t Bds3_B1c_Bgto::gnss_id {}
```
Definition at line 87 of file beidou_cnav1_navigation_message.h.

12.22.2.5 t0_bgto_s

```
int32_t Bds3_B1c_Bgto::t0_bgto_s {}
```
Definition at line 89 of file beidou_cnav1_navigation_message.h.

12.22.2.6 wn0_bgto

```
int32_t Bds3_B1c_Bgto::wn0_bgto {}
```
Definition at line 88 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.23 Bds3_B1c_Eop Struct Reference

Public Attributes

- int32_t t_eop_s {}
- double pm_x_arcsec {}
- double pm_x_dot_arcsec_day {}
- double pm_y_arcsec {}
- double pm_y_dot_arcsec_day {}
- double delta_ut1_s {}
- double delta_ut1_dot_s_day {}

12.23.1 Detailed Description

Definition at line 73 of file beidou_cnav1_navigation_message.h.

12.23.2 Member Data Documentation

12.23.2.1 delta_ut1_dot_s_day

```
double Bds3_B1c_Eop::delta_ut1_dot_s_day {}
```
Definition at line 81 of file beidou_cnav1_navigation_message.h.

12.23.2.2 delta_ut1_s

```
double Bds3_B1c_Eop::delta_ut1_s {}
```
Definition at line 80 of file beidou_cnav1_navigation_message.h.

12.23.2.3 pm_x_arcsec

```
double Bds3_B1c_Eop::pm_x_arcsec {}
```
Definition at line 76 of file beidou_cnav1_navigation_message.h.

12.23.2.4 pm_x_dot_arcsec_day

```
double Bds3_B1c_Eop::pm_x_dot_arcsec_day {}
```
Definition at line 77 of file beidou_cnav1_navigation_message.h.

12.23.2.5 pm_y_arcsec

```
double Bds3_B1c_Eop::pm_y_arcsec {}
```
Definition at line 78 of file beidou_cnav1_navigation_message.h.

12.23.2.6 pm_y_dot_arcsec_day

```
double Bds3_B1c_Eop::pm_y_dot_arcsec_day {}
```
Definition at line 79 of file beidou_cnav1_navigation_message.h.

12.23.2.7 t_eop_s

```
int32_t Bds3_B1c_Eop::t_eop_s {}
```
Definition at line 75 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.24 Bds3_B1c_PageCommon Struct Reference

Public Attributes

- int32_t page_id {}
- int32_t sismai {}
- int32_t hs {}
- bool dif {}
- bool sif {}
- bool aif {}

12.24.1 Detailed Description

Definition at line 32 of file beidou_cnav1_navigation_message.h.

12.24.2 Member Data Documentation

12.24.2.1 aif

```
bool Bds3_B1c_PageCommon::aif {}
```
Definition at line 39 of file beidou_cnav1_navigation_message.h.

12.24.2.2 dif

```
bool Bds3_B1c_PageCommon::dif {}
```
Definition at line 37 of file beidou_cnav1_navigation_message.h.

12.24.2.3 hs

```
int32_t Bds3_B1c_PageCommon::hs {}
```
Definition at line 36 of file beidou_cnav1_navigation_message.h.

12.24.2.4 page_id

```
int32_t Bds3_B1c_PageCommon::page_id {}
```
Definition at line 34 of file beidou_cnav1_navigation_message.h.

12.24.2.5 sif

```
bool Bds3_B1c_PageCommon::sif {}
```
Definition at line 38 of file beidou_cnav1_navigation_message.h.

12.24.2.6 sismai

```
int32_t Bds3_B1c_PageCommon::sismai {}
```

Definition at line 35 of file beidou_cnav1_navigation_message.h.

The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.25 Bds3_B1c_PageData Struct Reference

Public Attributes

- Bds3_B1c_PageCommon common {}
- int32_t sisaioc {}
- int32_t sisaioe {}
- int32_t wna {}
- int32_t toa_s {}
- std::array< Bds3_B1c_AlmanacReduced, 4 > reduced_almanac {}
- Bds3_B1c_Eop eop {}
- Bds3_B1c_Bgto bgto {}
- Bds3_B1c_AlmanacMedium medium_almanac {}

12.25.1 Detailed Description

Definition at line 96 of file beidou_cnav1_navigation_message.h.

12.25.2 Member Data Documentation

12.25.2.1 bgto

```
Bds3_B1c_Bgto Bds3_B1c_PageData::bgto {}
```

Definition at line 105 of file beidou_cnav1_navigation_message.h.

12.25.2.2 common

```
Bds3_B1c_PageCommon Bds3_B1c_PageData::common {}
```

Definition at line 98 of file beidou_cnav1_navigation_message.h.

12.25.2.3 eop

```
Bds3_B1c_Eop Bds3_B1c_PageData::eop {}
```

Definition at line 104 of file beidou_cnav1_navigation_message.h.

12.25.2.4 medium_almanac

```
Bds3_B1c_AlmanacMedium Bds3_B1c_PageData::medium_almanac {}
```

Definition at line 106 of file beidou_cnav1_navigation_message.h.

12.25.2.5 reduced_almanac

```
std::array<Bds3_B1c_AlmanacReduced, 4> Bds3_B1c_PageData::reduced_almanac {}
```

Definition at line 103 of file beidou_cnav1_navigation_message.h.

12.25.2.6 sisaioc

```
int32_t Bds3_B1c_PageData::sisaioc {}
```

Definition at line 99 of file beidou_cnav1_navigation_message.h.

12.25.2.7 sisaioe

`int32_t Bds3_B1c_PageData::sisaioe {}`
Definition at line 100 of file beidou_cnav1_navigation_message.h.

12.25.2.8 toa_s

`int32_t Bds3_B1c_PageData::toa_s {}`
Definition at line 102 of file beidou_cnav1_navigation_message.h.

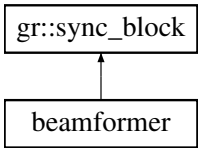
12.25.2.9 wna

`int32_t Bds3_B1c_PageData::wna {}`
Definition at line 101 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.26 beamformer Class Reference

This class implements a real-time software-defined spatial filter using the CTTC GNSS experimental antenna array input and a set of dynamically reloadable weights.
`#include <beamformer.h>`
Inheritance diagram for beamformer:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- beamformer_sptr **make_beamformer_sptr** ()

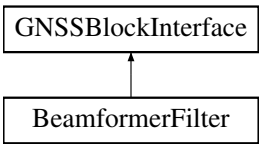
12.26.1 Detailed Description

This class implements a real-time software-defined spatial filter using the CTTC GNSS experimental antenna array input and a set of dynamically reloadable weights.
Definition at line 42 of file beamformer.h.
The documentation for this class was generated from the following file:

- beamformer.h

12.27 BeamformerFilter Class Reference

Interface of an adapter of a digital beamformer block to a GNSSBlockInterface.
`#include <beamformer_filter.h>`
Inheritance diagram for BeamformerFilter:



Public Member Functions

- **BeamformerFilter** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ stream, unsigned int out_stream)
- std::string role () override
- std::string implementation () override

  returns "Beamformer_Filte"
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.27.1 Detailed Description

Interface of an adapter of a digital beamformer block to a GNSSBlockInterface.
Definition at line 38 of file beamformer_filter.h.

12.27.2 Member Function Documentation

12.27.2.1 connect()

```
void BeamformerFilter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.27.2.2 disconnect()

```
void BeamformerFilter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.27.2.3 get_left_block()

```
gr::basic_block_sptr BeamformerFilter::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.27.2.4 get_right_block()

```
gr::basic_block_sptr BeamformerFilter::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.27.2.5 implementation()

`std::string BeamformerFilter::implementation () [inline], [override], [virtual]`
returns "Beamformer_Filte"
Implements GNSSBlockInterface.
Definition at line 53 of file beamformer_filter.h.

12.27.2.6 item_size()

`size_t BeamformerFilter::item_size () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 58 of file beamformer_filter.h.

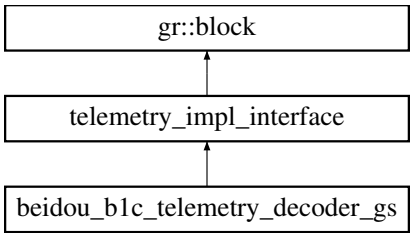
12.27.2.7 role()

`std::string BeamformerFilter::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 47 of file beamformer_filter.h.
The documentation for this class was generated from the following file:

- beamformer_filter.h

12.28 beidou_b1c_telemetry_decoder_gs Class Reference

Inheritance diagram for beidou_b1c_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override
- void set_channel (int channel) override
- void reset () override
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required) override
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)

Friends

- beidou_b1c_telemetry_decoder_gs_sptr **beidou_b1c_make_telemetry_decoder_gs** (const Gnss_Satellite &satellite, const Tlm_Conf &conf)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()

- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.28.1 Detailed Description

Definition at line 33 of file beidou_b1c_telemetry_decoder_gs.h.

12.28.2 Member Function Documentation

12.28.2.1 reset()

```
void beidou_b1c_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.28.2.2 set_channel()

```
void beidou_b1c_telemetry_decoder_gs::set_channel (
            int channel)  [override], [virtual]
```

Implements telemetry_impl_interface.

12.28.2.3 set_satellite()

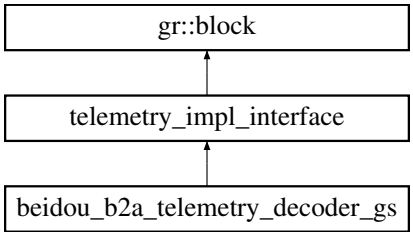
```
void beidou_b1c_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- beidou_b1c_telemetry_decoder_gs.h

12.29 beidou_b2a_telemetry_decoder_gs Class Reference

Inheritance diagram for beidou_b2a_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override
- void set_channel (int channel) override
- void reset () override
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required) override
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)

Friends

- beidou_b2a_telemetry_decoder_gs_sptr **beidou_b2a_make_telemetry_decoder_gs** (const Gnss_Satellite &satellite, const Tlm_Conf &conf)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.29.1 Detailed Description

Definition at line 36 of file beidou_b2a_telemetry_decoder_gs.h.

12.29.2 Member Function Documentation

12.29.2.1 reset()

```
void beidou_b2a_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.29.2.2 set_channel()

```
void beidou_b2a_telemetry_decoder_gs::set_channel (
            int channel)  [override], [virtual]
```

Implements telemetry_impl_interface.

12.29.2.3 set_satellite()

```
void beidou_b2a_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

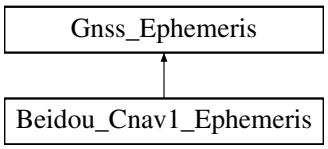
Implements telemetry_impl_interface.

The documentation for this class was generated from the following file:

- beidou_b2a_telemetry_decoder_gs.h

12.30 Beidou_Cnav1_Ephemeris Class Reference

Inheritance diagram for Beidou_Cnav1_Ephemeris:



Public Member Functions

- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

Public Member Functions inherited from Gnss_Ephemeris

- double sv_clock_drift (double transmitTime)

  Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- double TGD_B1Cp {}

  §7.6 B1C pilot group delay
- double TGD_B2ap {}

  §7.6 / B-CNAV2 MT30 B2a pilot group delay
- double ISC_B1Cd {}

  §7.6 B1C data-to-pilot intra-frequency delay correction
- double ISC_B2ad {}

  B-CNAV2 MT30 B2a data-to-pilot ISC.
- double IODC {}

  §7.4.2
- double IODE {}

  §7.4.1
- int32_t hs {}

  SF3 satellite health (2 bits, ICD §6.2.3).
- int32_t sig_type {BDS_EPH_SOURCE_CNAV1}

  RTKLIB eph.code for B-CNAV1.
- int32_t nav_type {1}

  0: GEO, 1: MEO/IGSO
- int32_t sat_type {}

  SF2 SatType (2 bits, ICD §7.5): 1=GEO, 2=IGSO, 3=MEO. Same encoding as RINEX 4 CNV1.

Public Attributes inherited from Gnss_Ephemeris

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

  Square root of the semi-major axis [meters^ 1/2].
- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
- double OMEGA_0 {}

Longitude of ascending node of orbital plane at weekly epoch [rad].

- double i_0 {}

  Inclination angle at reference time [rad].
- double omega {}

  Argument of perigee [rad].
- double OMEGAdot {}

  Rate of right ascension [rad/sec].
- double idot {}

  Rate of inclination angle [rad/sec].
- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].
- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].
- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].
- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].
- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].
- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].
- int32_t toe {}

  Ephemeris reference time [s].
- int32_t toc {}

  Clock correction data reference Time of Week [sec].
- double af0 {}

  SV clock bias correction coefficient [s].
- double af1 {}

  SV clock drift correction coefficient [s/s].
- double af2 {}

  SV clock drift rate correction coefficient [s/s^2].
- double satClkDrift {}

  SV clock drift.
- double dtr {}

  Relativistic clock correction term.
- int32_t WN {}

  Week number.
- int32_t tow {}

  Time of Week.
- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.
- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.
- double satpos_Z {}

  Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).
- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].
- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].
- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Additional Inherited Members

Protected Attributes inherited from Gnss_Ephemeris

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.30.1 Detailed Description

Definition at line 23 of file beidou_cnav1_ephemeris.h.

12.30.2 Constructor & Destructor Documentation

12.30.2.1 Beidou_Cnav1_Ephemeris()

```
Beidou_Cnav1_Ephemeris::Beidou_Cnav1_Ephemeris ()  [inline]
```

Definition at line 26 of file beidou_cnav1_ephemeris.h.

12.30.3 Member Function Documentation

12.30.3.1 serialize()

```
template<class Archive>
void Beidou_Cnav1_Ephemeris::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Definition at line 43 of file beidou_cnav1_ephemeris.h.

12.30.4 Member Data Documentation

12.30.4.1 hs

```
int32_t Beidou_Cnav1_Ephemeris::hs {}
```

SF3 satellite health (2 bits, ICD §6.2.3).
Definition at line 37 of file beidou_cnav1_ephemeris.h.

12.30.4.2 IODC

```
double Beidou_Cnav1_Ephemeris::IODC {}
```

§7.4.2
Definition at line 35 of file beidou_cnav1_ephemeris.h.

12.30.4.3 IODE

```
double Beidou_Cnav1_Ephemeris::IODE {}
```

§7.4.1
Definition at line 36 of file beidou_cnav1_ephemeris.h.

12.30.4.4 ISC_B1Cd

```
double Beidou_Cnav1_Ephemeris::ISC_B1Cd {}
```

§7.6 B1C data-to-pilot intra-frequency delay correction
Definition at line 33 of file beidou_cnav1_ephemeris.h.

12.30.4.5 ISC_B2ad

```
double Beidou_Cnav1_Ephemeris::ISC_B2ad {}
```

B-CNAV2 MT30 B2a data-to-pilot ISC.
Definition at line 34 of file beidou_cnav1_ephemeris.h.

12.30.4.6 nav_type

```
int32_t Beidou_Cnav1_Ephemeris::nav_type {1}
```

0: GEO, 1: MEO/IGSO
Definition at line 39 of file beidou_cnav1_ephemeris.h.

12.30.4.7 sat_type

```
int32_t Beidou_Cnav1_Ephemeris::sat_type {}
```

SF2 SatType (2 bits, ICD §7.5): 1=GEO, 2=IGSO, 3=MEO. Same encoding as RINEX 4 CNV1.
Definition at line 40 of file beidou_cnav1_ephemeris.h.

12.30.4.8 sig_type

```
int32_t Beidou_Cnav1_Ephemeris::sig_type {BDS_EPH_SOURCE_CNAV1}
```

RTKLIB eph.code for B-CNAV1.
Definition at line 38 of file beidou_cnav1_ephemeris.h.

12.30.4.9 TGD_B1Cp

```
double Beidou_Cnav1_Ephemeris::TGD_B1Cp {}
```

§7.6 B1C pilot group delay
Definition at line 31 of file beidou_cnav1_ephemeris.h.

12.30.4.10 TGD_B2ap

```
double Beidou_Cnav1_Ephemeris::TGD_B2ap {}
```

§7.6 / B-CNAV2 MT30 B2a pilot group delay
Definition at line 32 of file beidou_cnav1_ephemeris.h.
The documentation for this class was generated from the following file:

- beidou_cnav1_ephemeris.h

12.31 Beidou_Cnav1_Iono Class Reference

Public Attributes

- double alpha1 {}
- double alpha2 {}
- double alpha3 {}
- double alpha4 {}
- double alpha5 {}
- double alpha6 {}
- double alpha7 {}
- double alpha8 {}
- double alpha9 {}
- bool valid {}

12.31.1 Detailed Description

Definition at line 24 of file beidou_cnav1_iono.h.

12.31.2 Member Data Documentation

12.31.2.1 alpha1

```
double Beidou_Cnav1_Iono::alpha1 {}
```

Definition at line 27 of file beidou_cnav1_iono.h.

12.31.2.2 alpha2

```
double Beidou_Cnav1_Iono::alpha2 {}
```
Definition at line 28 of file beidou_cnav1_iono.h.

12.31.2.3 alpha3

```
double Beidou_Cnav1_Iono::alpha3 {}
```
Definition at line 29 of file beidou_cnav1_iono.h.

12.31.2.4 alpha4

```
double Beidou_Cnav1_Iono::alpha4 {}
```
Definition at line 30 of file beidou_cnav1_iono.h.

12.31.2.5 alpha5

```
double Beidou_Cnav1_Iono::alpha5 {}
```
Definition at line 31 of file beidou_cnav1_iono.h.

12.31.2.6 alpha6

```
double Beidou_Cnav1_Iono::alpha6 {}
```
Definition at line 32 of file beidou_cnav1_iono.h.

12.31.2.7 alpha7

```
double Beidou_Cnav1_Iono::alpha7 {}
```
Definition at line 33 of file beidou_cnav1_iono.h.

12.31.2.8 alpha8

```
double Beidou_Cnav1_Iono::alpha8 {}
```
Definition at line 34 of file beidou_cnav1_iono.h.

12.31.2.9 alpha9

```
double Beidou_Cnav1_Iono::alpha9 {}
```
Definition at line 35 of file beidou_cnav1_iono.h.

12.31.2.10 valid

```
bool Beidou_Cnav1_Iono::valid {}
```
Definition at line 36 of file beidou_cnav1_iono.h.
The documentation for this class was generated from the following file:

- beidou_cnav1_iono.h

12.32 Beidou_Cnav1_Navigation_Message Class Reference

Public Member Functions

- bool **decode_frame** (const float *symbols, int32_t num_symbols, int32_t expected_prn=0, int32_t *fail_↩ stage=nullptr)
- bool **decode_frame_symbols** (const float *symbols, int32_t num_symbols, int32_t expected_prn=0, int32↩ _t *fail_stage=nullptr)
- bool **probe_subframe1_prn** (const float *symbols, int32_t num_symbols, int32_t expected_prn) const
- bool **have_new_ephemeris** () const
- bool **have_new_iono** () const
- bool **have_new_utc_model** () const

- bool **have_new_page_data** () const
- void **clear_flags** ()
- const Beidou_Cnav1_Ephemeris & **get_ephemeris** () const
- const Beidou_Cnav1_Iono & **get_iono** () const
- const Beidou_Cnav1_Utc_Model & **get_utc_model** () const
- const Bds3_B1c_PageData & **get_page_data** () const
- const std::string & get_last_nav_bits () const

  Last successfully decoded frame as '0'/'1' chars for Nav_msg_from_TLM (SF1 info + SF2 [+ SF3]).
- double **get_tow_s** () const

12.32.1 Detailed Description

Definition at line 117 of file beidou_cnav1_navigation_message.h.

12.32.2 Member Function Documentation

12.32.2.1 get_last_nav_bits()

```
const std::string & Beidou_Cnav1_Navigation_Message::get_last_nav_bits () const
```

Last successfully decoded frame as '0'/'1' chars for Nav_msg_from_TLM (SF1 info + SF2 [+ SF3]).
The documentation for this class was generated from the following file:

- beidou_cnav1_navigation_message.h

12.33 Beidou_Cnav1_PageData_Message Struct Reference

Public Attributes

- int32_t PRN {}
- Bds3_B1c_PageData page_data {}

12.33.1 Detailed Description

Definition at line 110 of file beidou_cnav1_navigation_message.h.

12.33.2 Member Data Documentation

12.33.2.1 page_data

```
Bds3_B1c_PageData Beidou_Cnav1_PageData_Message::page_data {}
```

Definition at line 113 of file beidou_cnav1_navigation_message.h.

12.33.2.2 PRN

```
int32_t Beidou_Cnav1_PageData_Message::PRN {}
```

Definition at line 112 of file beidou_cnav1_navigation_message.h.
The documentation for this struct was generated from the following file:

- beidou_cnav1_navigation_message.h

12.34 Beidou_Cnav1_Utc_Model Class Reference

Public Attributes

- double A0 {}
- double A1 {}
- double A2 {}
- int32_t tot {}

- int32_t WN_t {}
- int32_t WN_LSF {}
- int32_t DN {}
- int32_t delta_t_LS {}

  Table 7-20: leap seconds before new leap effective [s].
- int32_t delta_t_LSF {}

  Table 7-20: leap seconds after new leap effective [s].
- bool valid {}

12.34.1 Detailed Description

Definition at line 26 of file beidou_cnav1_utc_model.h.

12.34.2 Member Data Documentation

12.34.2.1 A0

```
double Beidou_Cnav1_Utc_Model::A0 {}
```
Definition at line 29 of file beidou_cnav1_utc_model.h.

12.34.2.2 A1

```
double Beidou_Cnav1_Utc_Model::A1 {}
```
Definition at line 30 of file beidou_cnav1_utc_model.h.

12.34.2.3 A2

```
double Beidou_Cnav1_Utc_Model::A2 {}
```
Definition at line 31 of file beidou_cnav1_utc_model.h.

12.34.2.4 delta_t_LS

```
int32_t Beidou_Cnav1_Utc_Model::delta_t_LS {}
```
Table 7-20: leap seconds before new leap effective [s].
Definition at line 36 of file beidou_cnav1_utc_model.h.

12.34.2.5 delta_t_LSF

```
int32_t Beidou_Cnav1_Utc_Model::delta_t_LSF {}
```
Table 7-20: leap seconds after new leap effective [s].
Definition at line 37 of file beidou_cnav1_utc_model.h.

12.34.2.6 DN

```
int32_t Beidou_Cnav1_Utc_Model::DN {}
```
Definition at line 35 of file beidou_cnav1_utc_model.h.

12.34.2.7 tot

```
int32_t Beidou_Cnav1_Utc_Model::tot {}
```
Definition at line 32 of file beidou_cnav1_utc_model.h.

12.34.2.8 valid

```
bool Beidou_Cnav1_Utc_Model::valid {}
```
Definition at line 38 of file beidou_cnav1_utc_model.h.

12.34.2.9 WN_LSF

```
int32_t Beidou_Cnav1_Utc_Model::WN_LSF {}
```
Definition at line 34 of file beidou_cnav1_utc_model.h.

12.34.2.10 WN_t

```
int32_t Beidou_Cnav1_Utc_Model::WN_t {}
```
Definition at line 33 of file beidou_cnav1_utc_model.h.
The documentation for this class was generated from the following file:

- beidou_cnav1_utc_model.h

12.35 Beidou_Cnav2_Navigation_Message Class Reference

Public Member Functions

- bool **decode_frame_symbols** (const float *symbols, int32_t n_symbols, uint32_t prn)
- bool have_new_ephemeris () const
- Beidou_Cnav1_Ephemeris get_ephemeris () const
- void clear_flags ()
- void **reset** ()
- bool last_crc_ok () const
- int32_t last_mes_type () const
- int32_t last_sow () const
- uint32_t last_frame_prn () const
- const std::string & get_last_nav_bits () const

Static Public Member Functions

- static double preamble_detection_statistic (const float *current_1ms, const float *previous_1ms=nullptr)

  Soft preamble detection statistic on 1 ms prompt samples.

12.35.1 Detailed Description

Definition at line 30 of file beidou_cnav2_navigation_message.h.

12.35.2 Member Function Documentation

12.35.2.1 clear_flags()

```
void Beidou_Cnav2_Navigation_Message::clear_flags ()  [inline]
```
Definition at line 36 of file beidou_cnav2_navigation_message.h.

12.35.2.2 get_ephemeris()

```
Beidou_Cnav1_Ephemeris Beidou_Cnav2_Navigation_Message::get_ephemeris () const  [inline]
```
Definition at line 35 of file beidou_cnav2_navigation_message.h.

12.35.2.3 get_last_nav_bits()

```
const std::string & Beidou_Cnav2_Navigation_Message::get_last_nav_bits () const  [inline]
```
Definition at line 42 of file beidou_cnav2_navigation_message.h.

12.35.2.4 have_new_ephemeris()

```
bool Beidou_Cnav2_Navigation_Message::have_new_ephemeris () const  [inline]
```
Definition at line 34 of file beidou_cnav2_navigation_message.h.

12.35.2.5 last_crc_ok()

```
bool Beidou_Cnav2_Navigation_Message::last_crc_ok () const  [inline]
```
Definition at line 38 of file beidou_cnav2_navigation_message.h.

12.35.2.6 last_frame_prn()

```
uint32_t Beidou_Cnav2_Navigation_Message::last_frame_prn () const  [inline]
```
Definition at line 41 of file beidou_cnav2_navigation_message.h.

12.35.2.7 last_mes_type()

```
int32_t Beidou_Cnav2_Navigation_Message::last_mes_type () const  [inline]
```
Definition at line 39 of file beidou_cnav2_navigation_message.h.

12.35.2.8 last_sow()

```
int32_t Beidou_Cnav2_Navigation_Message::last_sow () const  [inline]
```
Definition at line 40 of file beidou_cnav2_navigation_message.h.

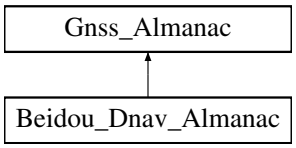
12.35.2.9 preamble_detection_statistic()

```
double Beidou_Cnav2_Navigation_Message::preamble_detection_statistic (
            const float * current_1ms,
            const float * previous_1ms = nullptr)  [static]
```
Soft preamble detection statistic on 1 ms prompt samples.
current_1ms points to BEIDOU_CNAV2_PREAMBLE_MS samples at a candidate frame start, previous_1ms (optional) to the samples one frame (3 s) earlier. With C the correlation with the preamble spread by the data secondary code and E the sample energy, it returns max(|C1| / sqrt(E1), |C1 + C2| / sqrt(E1 + E2)). The result does not depend on the input scale and is about |N(0,1)| on noise. The single-frame term keeps detection working across a carrier polarity flip between the two frames.
The documentation for this class was generated from the following file:

- beidou_cnav2_navigation_message.h

12.36 Beidou_Dnav_Almanac Class Reference

This class is a storage for the BeiDou D1/D2 almanac.
`#include <beidou_dnav_almanac.h>`
Inheritance diagram for Beidou_Dnav_Almanac:



Public Member Functions

- Beidou_Dnav_Almanac ()
- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

Public Member Functions inherited from Gnss_Almanac

- Gnss_Almanac ()=default
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

Computes prediction of the Doppler shift for a given time and receiver's position and velocity.

- void satellitePosVelComputation (double transmitTime, std::array< double, 7 > &pos_vel_dtr) const

  Computes satellite Position and Velocity, in ECEF, for a given time (expressed in seconds of week).

Public Attributes

- int SV_health {}

  SV Health.
- int AmEpID {}

  Identification of the expanded-almanac sequence.
- int AmID {}

  Identification of the expanded-almanac time-sharing group.
- bool expanded {}

  True for almanacs of SV ID 31 through 63.

Public Attributes inherited from Gnss_Almanac

- uint32_t PRN {}

  SV PRN NUMBER.
- double delta_i {}

  Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles).
- int32_t toa {}

  Almanac data reference time of week [s].
- int32_t WNa {}

  Almanac week number.
- double M_0 {}

  Mean Anomaly at Reference Time [semi-circles].
- double ecc {}

  Eccentricity [dimensionless].
- double sqrtA {}

  Square Root of the Semi-Major Axis [sqrt(m)].
- double OMEGA_0 {}

  Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].
- double omega {}

  Argument of Perigee [semi-cicles].
- double OMEGAdot {}

  Rate of Right Ascension [semi-circles/s].
- double af0 {}

  Coefficient 0 of code phase offset model [s].
- double af1 {}

  Coefficient 1 of code phase offset model [s/s].
- bool from_startup_load {false}

  Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the broadcast. Runtime-only provenance flag, intentionally not serialized.

Additional Inherited Members

Protected Attributes inherited from Gnss_Almanac

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J': QZSS.

12.36.1 Detailed Description

This class is a storage for the BeiDou D1/D2 almanac.
Definition at line 34 of file beidou_dnav_almanac.h.

12.36.2 Constructor & Destructor Documentation

12.36.2.1 Beidou_Dnav_Almanac()

```
Beidou_Dnav_Almanac::Beidou_Dnav_Almanac ()  [inline]
```

Default constructor
Definition at line 40 of file beidou_dnav_almanac.h.
References Gnss_Almanac::System.

12.36.3 Member Function Documentation

12.36.3.1 serialize()

```
template<class Archive>
void Beidou_Dnav_Almanac::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

Definition at line 52 of file beidou_dnav_almanac.h.

12.36.4 Member Data Documentation

12.36.4.1 AmEpID

```
int Beidou_Dnav_Almanac::AmEpID {}
```

Identification of the expanded-almanac sequence.
Definition at line 46 of file beidou_dnav_almanac.h.

12.36.4.2 AmID

```
int Beidou_Dnav_Almanac::AmID {}
```

Identification of the expanded-almanac time-sharing group.
Definition at line 47 of file beidou_dnav_almanac.h.

12.36.4.3 expanded

```
bool Beidou_Dnav_Almanac::expanded {}
```

True for almanacs of SV ID 31 through 63.
Definition at line 48 of file beidou_dnav_almanac.h.

12.36.4.4 SV_health

```
int Beidou_Dnav_Almanac::SV_health {}
```

SV Health.
Definition at line 45 of file beidou_dnav_almanac.h.
The documentation for this class was generated from the following file:

- beidou_dnav_almanac.h

12.37 Beidou_Dnav_Differential_Correction Struct Reference

Public Attributes

- uint32_t PRN {}
- int RURAI {}

  Regional User Range Accuracy Index.

- int UDREI {}

  User Differential Range Error Index.
- double delta_t {}

  Equivalent clock correction [m].
- bool delta_t_available {}

  False when the 13-bit value is -4096.

12.37.1 Detailed Description

Definition at line 28 of file beidou_dnav_differential_corrections.h.

12.37.2 Member Data Documentation

12.37.2.1 delta_t

```
double Beidou_Dnav_Differential_Correction::delta_t {}
```

Equivalent clock correction [m].
Definition at line 33 of file beidou_dnav_differential_corrections.h.

12.37.2.2 delta_t_available

```
bool Beidou_Dnav_Differential_Correction::delta_t_available {}
```

False when the 13-bit value is -4096.
Definition at line 34 of file beidou_dnav_differential_corrections.h.

12.37.2.3 PRN

```
uint32_t Beidou_Dnav_Differential_Correction::PRN {}
```

Definition at line 30 of file beidou_dnav_differential_corrections.h.

12.37.2.4 RURAI

```
int Beidou_Dnav_Differential_Correction::RURAI {}
```

Regional User Range Accuracy Index.
Definition at line 31 of file beidou_dnav_differential_corrections.h.

12.37.2.5 UDREI

```
int Beidou_Dnav_Differential_Correction::UDREI {}
```

User Differential Range Error Index.
Definition at line 32 of file beidou_dnav_differential_corrections.h.
The documentation for this struct was generated from the following file:

- beidou_dnav_differential_corrections.h

12.38 Beidou_Dnav_Differential_Corrections Class Reference

Public Attributes

- uint32_t source_PRN {}

  GEO satellite broadcasting this data set.
- double SOW {}
- int Pnum2 {}
- int SatH2 {}
- int BDEpID {}
- bool range_corrections_valid {}
- bool ionospheric_grid_valid {}
- bool expanded_corrections {}

- std::map< int, Beidou_Dnav_Differential_Correction > corrections

  Keyed by corrected SV ID.

12.38.1 Detailed Description

Definition at line 38 of file beidou_dnav_differential_corrections.h.

12.38.2 Member Data Documentation

12.38.2.1 BDEpID

```
int Beidou_Dnav_Differential_Corrections::BDEpID {}
```

Definition at line 45 of file beidou_dnav_differential_corrections.h.

12.38.2.2 corrections

```
std::map<int, Beidou_Dnav_Differential_Correction> Beidou_Dnav_Differential_Corrections::↩
corrections
```

Keyed by corrected SV ID.
Definition at line 49 of file beidou_dnav_differential_corrections.h.

12.38.2.3 expanded_corrections

```
bool Beidou_Dnav_Differential_Corrections::expanded_corrections {}
```

Definition at line 48 of file beidou_dnav_differential_corrections.h.

12.38.2.4 ionospheric_grid_valid

```
bool Beidou_Dnav_Differential_Corrections::ionospheric_grid_valid {}
```

Definition at line 47 of file beidou_dnav_differential_corrections.h.

12.38.2.5 Pnum2

```
int Beidou_Dnav_Differential_Corrections::Pnum2 {}
```

Definition at line 43 of file beidou_dnav_differential_corrections.h.

12.38.2.6 range_corrections_valid

```
bool Beidou_Dnav_Differential_Corrections::range_corrections_valid {}
```

Definition at line 46 of file beidou_dnav_differential_corrections.h.

12.38.2.7 SatH2

```
int Beidou_Dnav_Differential_Corrections::SatH2 {}
```

Definition at line 44 of file beidou_dnav_differential_corrections.h.

12.38.2.8 source_PRN

```
uint32_t Beidou_Dnav_Differential_Corrections::source_PRN {}
```

GEO satellite broadcasting this data set.
Definition at line 41 of file beidou_dnav_differential_corrections.h.

12.38.2.9 SOW

```
double Beidou_Dnav_Differential_Corrections::SOW {}
```

Definition at line 42 of file beidou_dnav_differential_corrections.h.
The documentation for this class was generated from the following file:

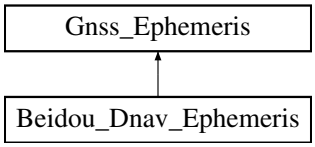
- beidou_dnav_differential_corrections.h

12.39 Beidou_Dnav_Ephemeris Class Reference

This is a storage class for the Beidou SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0).

`#include <beidou_dnav_ephemeris.h>`

Inheritance diagram for Beidou_Dnav_Ephemeris:



Public Member Functions

- Beidou_Dnav_Ephemeris ()
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Member Functions inherited from Gnss_Ephemeris

- double sv_clock_drift (double transmitTime)

  Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- int SV_accuracy {}

  User Range Accuracy (URA) index of the SV (reference paragraph 5.2.4.5) for the standard positioning service user.
- int SV_health {}
- double TGD1 {}

  Estimated Group Delay Differential on B1I [s].
- double TGD2 {}

  Estimated Group Delay Differential on B2I [s].
- double AODC {}

  Age of Data, Clock.
- double AODE {}

  Age of Data, Ephemeris.
- int AODO {}

  Age of Data Offset (AODO) term for the navigation message correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s].
- int sig_type {}

  BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q) ∗/.
- int nav_type {}

  BDS: nav type (0:unknown,1:IGSO/MEO,2:GEO) ∗/.
- bool fit_interval_flag {}

  Curve-fit interval used by the CS (Block II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4 hours, 1 = greater than 4 hours.

- double spare1 {}
- double spare2 {}
- bool integrity_status_flag {}

  If true, enhanced level of integrity assurance.
- bool alert_flag {}

  If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
- bool antispoofing_flag {}

  If true, the AntiSpoofing mode is ON in that SV.
- std::map< int, std::string > satelliteBlock

  Map that stores to which block the PRN belongs.

Public Attributes inherited from Gnss_Ephemeris

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

  Square root of the semi-major axis [meters^ 1/2].
- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
- double OMEGA_0 {}

  Longitude of ascending node of orbital plane at weekly epoch [rad].
- double i_0 {}

  Inclination angle at reference time [rad].
- double omega {}

  Argument of perigee [rad].
- double OMEGAdot {}

  Rate of right ascension [rad/sec].
- double idot {}

  Rate of inclination angle [rad/sec].
- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].
- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].
- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].
- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].
- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].
- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].

- int32_t toe {}

  Ephemeris reference time [s].
- int32_t toc {}

  Clock correction data reference Time of Week [sec].
- double af0 {}

  SV clock bias correction coefficient [s].
- double af1 {}

  SV clock drift correction coefficient [s/s].
- double af2 {}

  SV clock drift rate correction coefficient [s/s^ 2].
- double satClkDrift {}

  SV clock drift.
- double dtr {}

  Relativistic clock correction term.
- int32_t WN {}

  Week number.
- int32_t tow {}

  Time of Week.
- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.
- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.
- double satpos_Z {}

  Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).
- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].
- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].
- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Additional Inherited Members

Protected Attributes inherited from Gnss_Ephemeris

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.39.1 Detailed Description

This is a storage class for the Beidou SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0).
See http://en.beidou.gov.cn/SYSTEMS/Officialdocument/201902/P020190227601370045731.↩ pdf
Definition at line 39 of file beidou_dnav_ephemeris.h.

12.39.2 Constructor & Destructor Documentation

12.39.2.1 Beidou_Dnav_Ephemeris()

```
Beidou_Dnav_Ephemeris::Beidou_Dnav_Ephemeris ()
```

Default constructor

12.39.3 Member Function Documentation

12.39.3.1 serialize()

```
template<class Archive>
void Beidou_Dnav_Ephemeris::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.
Definition at line 87 of file beidou_dnav_ephemeris.h.
References Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, alert_flag, antispoofing_flag, AODC, AODE, AODO, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::dtr, Gnss_Ephemeris::ecc, fit_interval_flag, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, integrity_status_flag, Gnss_Ephemeris::M_0, nav_type, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdot, Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, sig_type, Gnss_Ephemeris::sqrtA, SV_accuracy, TGD1, TGD2, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, Gnss_Ephemeris::tow, and Gnss_Ephemeris::WN.

12.39.4 Member Data Documentation

12.39.4.1 alert_flag

```
bool Beidou_Dnav_Ephemeris::alert_flag {}
```

If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
Definition at line 76 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.2 antispoofing_flag

```
bool Beidou_Dnav_Ephemeris::antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.
Definition at line 77 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.3 AODC

```
double Beidou_Dnav_Ephemeris::AODC {}
```

Age of Data, Clock.
Definition at line 51 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.4 AODE

```
double Beidou_Dnav_Ephemeris::AODE {}
```

Age of Data, Ephemeris.
Definition at line 52 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.5 AODO

```
int Beidou_Dnav_Ephemeris::AODO {}
```

Age of Data Offset (AODO) term for the navigation message correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s].
Definition at line 53 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.6 fit_interval_flag

```
bool Beidou_Dnav_Ephemeris::fit_interval_flag {}
```

Curve-fit interval used by the CS (Block II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4 hours, 1 = greater than 4 hours.
Definition at line 58 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.7 integrity_status_flag

```
bool Beidou_Dnav_Ephemeris::integrity_status_flag {}
```

If true, enhanced level of integrity assurance.
If false, indicates that the conveying signal is provided with the legacy level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 4.42 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-5 per hour.
If true, indicates that the conveying signal is provided with an enhanced level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 5.73 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-8 per hour.
Definition at line 75 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.8 nav_type

```
int Beidou_Dnav_Ephemeris::nav_type {}
```

BDS: nav type (0:unknown,1:IGSO/MEO,2:GEO) */.
Definition at line 56 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.9 satelliteBlock

```
std::map<int, std::string> Beidou_Dnav_Ephemeris::satelliteBlock
```

Map that stores to which block the PRN belongs.
Definition at line 79 of file beidou_dnav_ephemeris.h.

12.39.4.10 sig_type

```
int Beidou_Dnav_Ephemeris::sig_type {}
```

BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q) */.
Definition at line 55 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.11 spare1

```
double Beidou_Dnav_Ephemeris::spare1 {}
```

Definition at line 59 of file beidou_dnav_ephemeris.h.

12.39.4.12 spare2

```
double Beidou_Dnav_Ephemeris::spare2 {}
```

Definition at line 60 of file beidou_dnav_ephemeris.h.

12.39.4.13 SV_accuracy

```
int Beidou_Dnav_Ephemeris::SV_accuracy {}
```

User Range Accuracy (URA) index of the SV (reference paragraph 5.2.4.5) for the standard positioning service user.
Definition at line 47 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

12.39.4.14 SV_health

```
int Beidou_Dnav_Ephemeris::SV_health {}
```

Definition at line 48 of file beidou_dnav_ephemeris.h.

12.39.4.15 TGD1

`double Beidou_Dnav_Ephemeris::TGD1 {}`
Estimated Group Delay Differential on B1I [s].
Definition at line 49 of file beidou_dnav_ephemeris.h.
Referenced by serialize().

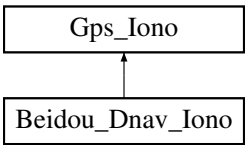
12.39.4.16 TGD2

`double Beidou_Dnav_Ephemeris::TGD2 {}`
Estimated Group Delay Differential on B2I [s].
Definition at line 50 of file beidou_dnav_ephemeris.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- beidou_dnav_ephemeris.h

12.40 Beidou_Dnav_Iono Class Reference

This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1.
`#include <beidou_dnav_iono.h>`
Inheritance diagram for Beidou_Dnav_Iono:



Public Member Functions

- Beidou_Dnav_Iono ()=default

  Default constructor.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

Public Member Functions inherited from Gps_Iono

- Gps_Iono ()=default

  Default constructor.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Attributes

- std::map< int, Beidou_Dnav_Iono_Grid_Point > grid_points

  D2 grid corrections, keyed by IGP number.
- bool grid_valid {}

Public Attributes inherited from Gps_Iono

- double alpha0 {}

  Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
- double alpha1 {}

Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].

- double alpha2 {}

  Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].

- double alpha3 {}

  Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].

- double beta0 {}

  Coefficient 0 of a cubic equation representing the period of the model [s].

- double beta1 {}

  Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].

- double beta2 {}

  Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].

- double beta3 {}

  Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].

- bool valid {}

  Valid flag.

12.40.1 Detailed Description

This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1.
Definition at line 58 of file beidou_dnav_iono.h.

12.40.2 Constructor & Destructor Documentation

12.40.2.1 Beidou_Dnav_Iono()

```
Beidou_Dnav_Iono::Beidou_Dnav_Iono ()  [default]
```

Default constructor.

12.40.3 Member Function Documentation

12.40.3.1 serialize()

```
template<class Archive>
void Beidou_Dnav_Iono::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Definition at line 67 of file beidou_dnav_iono.h.

12.40.4 Member Data Documentation

12.40.4.1 grid_points

```
std::map<int, Beidou_Dnav_Iono_Grid_Point> Beidou_Dnav_Iono::grid_points
```

D2 grid corrections, keyed by IGP number.
Definition at line 63 of file beidou_dnav_iono.h.

12.40.4.2 grid_valid

```
bool Beidou_Dnav_Iono::grid_valid {}
```

Definition at line 64 of file beidou_dnav_iono.h.
The documentation for this class was generated from the following file:

- beidou_dnav_iono.h

12.41 Beidou_Dnav_Iono_Grid_Point Struct Reference

Public Member Functions

- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

Public Attributes

- double vertical_delay {}

  B1I vertical delay [m].
- int GIVEI {}

  Grid Ionospheric Vertical Error Index.
- bool monitored {}

  False when the broadcast delay code is 510.
- bool available {}

  False when the broadcast delay code is 511.

12.41.1 Detailed Description

Definition at line 28 of file beidou_dnav_iono.h.

12.41.2 Member Function Documentation

12.41.2.1 serialize()

```
template<class Archive>
void Beidou_Dnav_Iono_Grid_Point::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Definition at line 36 of file beidou_dnav_iono.h.

12.41.3 Member Data Documentation

12.41.3.1 available

```
bool Beidou_Dnav_Iono_Grid_Point::available {}
```

False when the broadcast delay code is 511.
Definition at line 33 of file beidou_dnav_iono.h.

12.41.3.2 GIVEI

```
int Beidou_Dnav_Iono_Grid_Point::GIVEI {}
```

Grid Ionospheric Vertical Error Index.
Definition at line 31 of file beidou_dnav_iono.h.

12.41.3.3 monitored

```
bool Beidou_Dnav_Iono_Grid_Point::monitored {}
```

False when the broadcast delay code is 510.
Definition at line 32 of file beidou_dnav_iono.h.

12.41.3.4 vertical_delay

```
double Beidou_Dnav_Iono_Grid_Point::vertical_delay {}
```

B1I vertical delay [m].
Definition at line 30 of file beidou_dnav_iono.h.
The documentation for this struct was generated from the following file:

- beidou_dnav_iono.h

12.42 Beidou_Dnav_Navigation_Message Class Reference

This class decodes BeiDou D1 and D2 NAV data messages.
`#include <beidou_dnav_navigation_message.h>`

Public Member Functions

- Beidou_Dnav_Navigation_Message ()
- Beidou_Dnav_Ephemeris get_ephemeris () const

  Obtain a BDS SV Ephemeris class filled with current SV data.
- Beidou_Dnav_Iono get_iono ()

  Obtain a BDS ionospheric correction parameters class filled with current SV data.
- Beidou_Dnav_Utc_Model get_utc_model ()

  Obtain a BDS UTC model parameters class filled with current SV data.
- Beidou_Dnav_Almanac get_almanac () const

  Obtain the last decoded BeiDou almanac record.
- Beidou_Dnav_Differential_Corrections get_differential_corrections () const

  Obtain the latest D2 integrity and differential corrections.
- int32_t d1_subframe_decoder (std::string const &subframe)

  Decodes the BDS D1 NAV message.
- int32_t d2_subframe_decoder (std::string const &subframe)

  Decodes the BDS D2 NAV message.
- double utc_time (double beidoutime_corrected) const

  Computes the Coordinated Universal Time (UTC) and returns it in [s].
- bool **satellite_validation** ()
- bool have_new_ephemeris ()

  Returns true if new Ephemeris has arrived. The flag is set to false when the function is executed.
- bool have_new_iono () const

  Returns true if new Iono model has arrived. The flag is set to false when the function is executed.
- bool have_new_utc_model ()

  Returns true if new UTC model has arrived. The flag is set to false when the function is executed.
- bool have_new_almanac ()

  Returns true if new UTC model has arrived. The flag is set to false when the function is executed.
- bool have_new_differential_corrections ()

  Returns true when a complete D2 subframes 2-4 correction page has arrived.
- void set_satellite_PRN (uint32_t prn)

  Sets satellite PRN number.
- void set_signal_type (int32_t signal_type)
- bool get_flag_CRC_test () const
- bool get_flag_new_SOW_available () const
- void set_flag_new_SOW_available (bool new_SOW_available)
- double get_SOW () const

12.42.1 Detailed Description

This class decodes BeiDou D1 and D2 NAV data messages.
Definition at line 50 of file beidou_dnav_navigation_message.h.

12.42.2 Constructor & Destructor Documentation

12.42.2.1 Beidou_Dnav_Navigation_Message()

`Beidou_Dnav_Navigation_Message::Beidou_Dnav_Navigation_Message ()`
Default constructor

12.42.3 Member Function Documentation

12.42.3.1 d1_subframe_decoder()

```
int32_t Beidou_Dnav_Navigation_Message::d1_subframe_decoder (
            std::string const & subframe)
```

Decodes the BDS D1 NAV message.

12.42.3.2 d2_subframe_decoder()

```
int32_t Beidou_Dnav_Navigation_Message::d2_subframe_decoder (
            std::string const & subframe)
```

Decodes the BDS D2 NAV message.

12.42.3.3 get_almanac()

```
Beidou_Dnav_Almanac Beidou_Dnav_Navigation_Message::get_almanac () const
```

Obtain the last decoded BeiDou almanac record.

12.42.3.4 get_differential_corrections()

```
Beidou_Dnav_Differential_Corrections Beidou_Dnav_Navigation_Message::get_differential_corrections
() const
```

Obtain the latest D2 integrity and differential corrections.

12.42.3.5 get_ephemeris()

```
Beidou_Dnav_Ephemeris Beidou_Dnav_Navigation_Message::get_ephemeris () const
```

Obtain a BDS SV Ephemeris class filled with current SV data.

12.42.3.6 get_flag_CRC_test()

```
bool Beidou_Dnav_Navigation_Message::get_flag_CRC_test () const  [inline]
```

Definition at line 139 of file beidou_dnav_navigation_message.h.

12.42.3.7 get_flag_new_SOW_available()

```
bool Beidou_Dnav_Navigation_Message::get_flag_new_SOW_available () const  [inline]
```

Definition at line 144 of file beidou_dnav_navigation_message.h.

12.42.3.8 get_iono()

```
Beidou_Dnav_Iono Beidou_Dnav_Navigation_Message::get_iono ()
```

Obtain a BDS ionospheric correction parameters class filled with current SV data.

12.42.3.9 get_SOW()

```
double Beidou_Dnav_Navigation_Message::get_SOW () const  [inline]
```

Definition at line 154 of file beidou_dnav_navigation_message.h.

12.42.3.10 get_utc_model()

```
Beidou_Dnav_Utc_Model Beidou_Dnav_Navigation_Message::get_utc_model ()
```

Obtain a BDS UTC model parameters class filled with current SV data.

12.42.3.11 have_new_almanac()

```
bool Beidou_Dnav_Navigation_Message::have_new_almanac ()
```

Returns true if new UTC model has arrived. The flag is set to false when the function is executed.

12.42.3.12 have_new_differential_corrections()

```
bool Beidou_Dnav_Navigation_Message::have_new_differential_corrections ()
```
Returns true when a complete D2 subframes 2-4 correction page has arrived.

12.42.3.13 have_new_ephemeris()

```
bool Beidou_Dnav_Navigation_Message::have_new_ephemeris ()
```
Returns true if new Ephemeris has arrived. The flag is set to false when the function is executed.

12.42.3.14 have_new_iono()

```
bool Beidou_Dnav_Navigation_Message::have_new_iono () const
```
Returns true if new Iono model has arrived. The flag is set to false when the function is executed.

12.42.3.15 have_new_utc_model()

```
bool Beidou_Dnav_Navigation_Message::have_new_utc_model ()
```
Returns true if new UTC model has arrived. The flag is set to false when the function is executed.

12.42.3.16 set_flag_new_SOW_available()

```
void Beidou_Dnav_Navigation_Message::set_flag_new_SOW_available (
            bool new_SOW_available)  [inline]
```
Definition at line 149 of file beidou_dnav_navigation_message.h.

12.42.3.17 set_satellite_PRN()

```
void Beidou_Dnav_Navigation_Message::set_satellite_PRN (
            uint32_t prn)  [inline]
```
Sets satellite PRN number.
Definition at line 129 of file beidou_dnav_navigation_message.h.

12.42.3.18 set_signal_type()

```
void Beidou_Dnav_Navigation_Message::set_signal_type (
            int32_t signal_type)  [inline]
```
Definition at line 134 of file beidou_dnav_navigation_message.h.

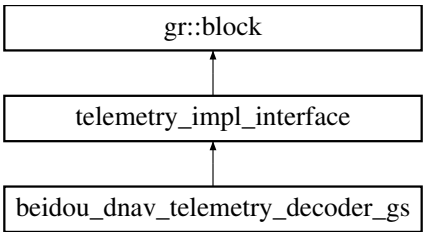
12.42.3.19 utc_time()

```
double Beidou_Dnav_Navigation_Message::utc_time (
            double beidoutime_corrected) const
```
Computes the Coordinated Universal Time (UTC) and returns it in [s].
The documentation for this class was generated from the following file:

- beidou_dnav_navigation_message.h

12.43 beidou_dnav_telemetry_decoder_gs Class Reference

This class implements a block that decodes the BeiDou DNAV data, broadcast by both the B1I and B3I signals (frequency_band = 1 or 3, respectively).
`#include <beidou_dnav_telemetry_decoder_gs.h>`
Inheritance diagram for beidou_dnav_telemetry_decoder_gs:



Public Member Functions

- ~beidou_dnav_telemetry_decoder_gs () override

  Class destructor.
- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int channel) override

  Set receiver's channel.
- void reset () override
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)

Friends

- beidou_dnav_telemetry_decoder_gs_sptr **beidou_dnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.43.1 Detailed Description

This class implements a block that decodes the BeiDou DNAV data, broadcast by both the B1I and B3I signals (frequency_band = 1 or 3, respectively).

Note

Code added as part of GSoC 2018 program

Definition at line 51 of file beidou_dnav_telemetry_decoder_gs.h.

12.43.2 Constructor & Destructor Documentation

12.43.2.1 ~beidou_dnav_telemetry_decoder_gs()

```
beidou_dnav_telemetry_decoder_gs::~beidou_dnav_telemetry_decoder_gs ()  [override]
```

Class destructor.

12.43.3 Member Function Documentation

12.43.3.1 general_work()

```
int beidou_dnav_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.43.3.2 reset()

```
void beidou_dnav_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.43.3.3 set_channel()

```
void beidou_dnav_telemetry_decoder_gs::set_channel (
            int channel)  [override], [virtual]
```

Set receiver's channel.
Implements telemetry_impl_interface.

12.43.3.4 set_satellite()

```
void beidou_dnav_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- beidou_dnav_telemetry_decoder_gs.h

12.44 Beidou_Dnav_Utc_Model Class Reference

This class is a storage for the BeiDou DNAV UTC Model.
`#include <beidou_dnav_utc_model.h>`

Public Member Functions

- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

Public Attributes

- double A0_UTC {}

  BDT clock bias relative to UTC [s].
- double A1_UTC {}

  BDT clock rate relative to UTC [s/s].
- int DeltaT_LS {}

  Delta time due to leap seconds before the new leap second effective.
- int WN_LSF {}

  Week number of the new leap second.
- int DN {}

  Day number of week of the new leap second.
- double DeltaT_LSF {}

  Delta time due to leap seconds after the new leap second effective [s].
- double A0_GPS {}

BDT clock bias relative to GPS time [s].

- double A1_GPS {}

  BDT clock rate relative to GPS time [s/s].

- double A0_GAL {}

  BDT clock bias relative to GAL time [s].

- double A1_GAL {}

  BDT clock rate relative to GAL time [s/s].

- double A0_GLO {}

  BDT clock bias relative to GLO time [s].

- double A1_GLO {}

  BDT clock rate relative to GLO time [s/s].

- bool valid {}

12.44.1 Detailed Description

This class is a storage for the BeiDou DNAV UTC Model.
Implementation follows the interface described in the Open Service Signal (Version 2.1)
Definition at line 35 of file beidou_dnav_utc_model.h.

12.44.2 Member Function Documentation

12.44.2.1 serialize()

```
template<class Archive>
void Beidou_Dnav_Utc_Model::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Definition at line 67 of file beidou_dnav_utc_model.h.

12.44.3 Member Data Documentation

12.44.3.1 A0_GAL

```
double Beidou_Dnav_Utc_Model::A0_GAL {}
```

BDT clock bias relative to GAL time [s].
Definition at line 53 of file beidou_dnav_utc_model.h.

12.44.3.2 A0_GLO

```
double Beidou_Dnav_Utc_Model::A0_GLO {}
```

BDT clock bias relative to GLO time [s].
Definition at line 57 of file beidou_dnav_utc_model.h.

12.44.3.3 A0_GPS

```
double Beidou_Dnav_Utc_Model::A0_GPS {}
```

BDT clock bias relative to GPS time [s].
Definition at line 49 of file beidou_dnav_utc_model.h.

12.44.3.4 A0_UTC

```
double Beidou_Dnav_Utc_Model::A0_UTC {}
```

BDT clock bias relative to UTC [s].
Definition at line 41 of file beidou_dnav_utc_model.h.

12.44.3.5 A1_GAL

`double Beidou_Dnav_Utc_Model::A1_GAL {}`
BDT clock rate relative to GAL time [s/s].
Definition at line 54 of file beidou_dnav_utc_model.h.

12.44.3.6 A1_GLO

`double Beidou_Dnav_Utc_Model::A1_GLO {}`
BDT clock rate relative to GLO time [s/s].
Definition at line 58 of file beidou_dnav_utc_model.h.

12.44.3.7 A1_GPS

`double Beidou_Dnav_Utc_Model::A1_GPS {}`
BDT clock rate relative to GPS time [s/s].
Definition at line 50 of file beidou_dnav_utc_model.h.

12.44.3.8 A1_UTC

`double Beidou_Dnav_Utc_Model::A1_UTC {}`
BDT clock rate relative to UTC [s/s].
Definition at line 42 of file beidou_dnav_utc_model.h.

12.44.3.9 DeltaT_LS

`int Beidou_Dnav_Utc_Model::DeltaT_LS {}`
Delta time due to leap seconds before the new leap second effective.
Definition at line 43 of file beidou_dnav_utc_model.h.

12.44.3.10 DeltaT_LSF

`double Beidou_Dnav_Utc_Model::DeltaT_LSF {}`
Delta time due to leap seconds after the new leap second effective [s].
Definition at line 46 of file beidou_dnav_utc_model.h.

12.44.3.11 DN

`int Beidou_Dnav_Utc_Model::DN {}`
Day number of week of the new leap second.
Definition at line 45 of file beidou_dnav_utc_model.h.

12.44.3.12 valid

`bool Beidou_Dnav_Utc_Model::valid {}`
Definition at line 60 of file beidou_dnav_utc_model.h.

12.44.3.13 WN_LSF

`int Beidou_Dnav_Utc_Model::WN_LSF {}`
Week number of the new leap second.
Definition at line 44 of file beidou_dnav_utc_model.h.
The documentation for this class was generated from the following file:

- beidou_dnav_utc_model.h

12.45 BeidouLdpcGraph Struct Reference

Public Attributes

- int32_t num_checks = 0
- int32_t num_variables = 0
- int32_t row_weight = 0
- std::vector< uint32_t > check_offsets
- std::vector< uint16_t > check_to_var
- std::vector< uint8_t > check_to_h
- std::vector< uint32_t > var_offsets
- std::vector< uint16_t > var_to_check
- std::vector< uint32_t > var_to_edge
- std::vector< uint8_t > var_to_h
- std::vector< uint8_t > var_to_h_inv

12.45.1 Detailed Description

Definition at line 86 of file beidou_ldpc.h.

12.45.2 Member Data Documentation

12.45.2.1 check_offsets

```
std::vector<uint32_t> BeidouLdpcGraph::check_offsets
```

Definition at line 93 of file beidou_ldpc.h.

12.45.2.2 check_to_h

```
std::vector<uint8_t> BeidouLdpcGraph::check_to_h
```

Definition at line 95 of file beidou_ldpc.h.

12.45.2.3 check_to_var

```
std::vector<uint16_t> BeidouLdpcGraph::check_to_var
```

Definition at line 94 of file beidou_ldpc.h.

12.45.2.4 num_checks

```
int32_t BeidouLdpcGraph::num_checks = 0
```

Definition at line 88 of file beidou_ldpc.h.

12.45.2.5 num_variables

```
int32_t BeidouLdpcGraph::num_variables = 0
```

Definition at line 89 of file beidou_ldpc.h.

12.45.2.6 row_weight

```
int32_t BeidouLdpcGraph::row_weight = 0
```

Definition at line 90 of file beidou_ldpc.h.

12.45.2.7 var_offsets

```
std::vector<uint32_t> BeidouLdpcGraph::var_offsets
```

Definition at line 98 of file beidou_ldpc.h.

12.45.2.8 var_to_check

`std::vector<uint16_t> BeidouLdpcGraph::var_to_check`
Definition at line 99 of file beidou_ldpc.h.

12.45.2.9 var_to_edge

`std::vector<uint32_t> BeidouLdpcGraph::var_to_edge`
Definition at line 100 of file beidou_ldpc.h.

12.45.2.10 var_to_h

`std::vector<uint8_t> BeidouLdpcGraph::var_to_h`
Definition at line 101 of file beidou_ldpc.h.

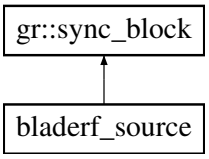
12.45.2.11 var_to_h_inv

`std::vector<uint8_t> BeidouLdpcGraph::var_to_h_inv`
Definition at line 102 of file beidou_ldpc.h.
The documentation for this struct was generated from the following file:

- beidou_ldpc.h

12.46 bladerf_source Class Reference

This class implements a GNU Radio source block that streams RX samples from a bladeRF device (single RX channel) using libbladeRF's synchronous streaming API, converting the device's native BLADERF_FORMAT_↩ SC16_Q11 samples to gr_complex.
`#include <bladerf_source.h>`
Inheritance diagram for bladerf_source:



Public Member Functions

- bool **start** () override
- bool **stop** () override
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

Friends

- bladerf_source_sptr **bladerf_make_source_sptr** (const std::string &bladerf_args, const std::string &device_serial, double sample_rate, double freq, double bandwidth, double gain, bool agc_enabled, bool rx_bias_tee, unsigned int num_buffers, unsigned int buffer_size, unsigned int num_transfers, unsigned int stream_timeout_ms)

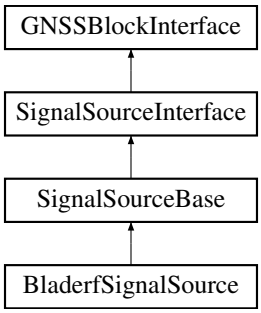
12.46.1 Detailed Description

This class implements a GNU Radio source block that streams RX samples from a bladeRF device (single RX channel) using libbladeRF's synchronous streaming API, converting the device's native BLADERF_FORMAT_↩ SC16_Q11 samples to gr_complex.
Definition at line 59 of file bladerf_source.h.
The documentation for this class was generated from the following file:

- bladerf_source.h

12.47 BladerfSignalSource Class Reference

This class instantiates the bladeRF signal source, streaming RX samples directly through libbladeRF (no gr-osmosdr or other intermediate driver required).
`#include <bladerf_signal_source.h>`
Inheritance diagram for BladerfSignalSource:



Public Member Functions

- **BladerfSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.47.1 Detailed Description

This class instantiates the bladeRF signal source, streaming RX samples directly through libbladeRF (no gr-osmosdr or other intermediate driver required).
Definition at line 42 of file bladerf_signal_source.h.

12.47.2 Member Function Documentation

12.47.2.1 connect()

```
void BladerfSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.47.2.2 disconnect()

```
void BladerfSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.47.2.3 get_left_block()

```
gr::basic_block_sptr BladerfSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.47.2.4 get_right_block()

```
gr::basic_block_sptr BladerfSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.47.2.5 item_size()

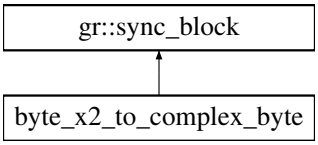
```
size_t BladerfSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 51 of file bladerf_signal_source.h.
The documentation for this class was generated from the following file:

- bladerf_signal_source.h

12.48 byte_x2_to_complex_byte Class Reference

This class adapts two signed char streams into a std::complex<signed char> stream.
`#include <byte_x2_to_complex_byte.h>`
Inheritance diagram for byte_x2_to_complex_byte:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- byte_x2_to_complex_byte_sptr **make_byte_x2_to_complex_byte** ()

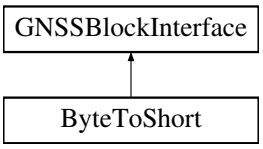
12.48.1 Detailed Description

This class adapts two signed char streams into a std::complex<signed char> stream.
Definition at line 41 of file byte_x2_to_complex_byte.h.
The documentation for this class was generated from the following file:

- byte_x2_to_complex_byte.h

12.49 ByteToShort Class Reference

Adapts an 8-bits sample stream (IF) to a short int stream (IF).
`#include <byte_to_short.h>`
Inheritance diagram for ByteToShort:



Public Member Functions

- **ByteToShort** (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Byte_To_Short".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.49.1 Detailed Description

Adapts an 8-bits sample stream (IF) to a short int stream (IF).
Definition at line 40 of file byte_to_short.h.

12.49.2 Member Function Documentation

12.49.2.1 connect()

```
void ByteToShort::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.49.2.2 disconnect()

```
void ByteToShort::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.49.2.3 get_left_block()

```
gr::basic_block_sptr ByteToShort::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.49.2.4 get_right_block()

`gr::basic_block_sptr ByteToShort::get_right_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.49.2.5 implementation()

`std::string ByteToShort::implementation () [inline], [override], [virtual]`
Returns "Byte_To_Short".
Implements GNSSBlockInterface.
Definition at line 55 of file byte_to_short.h.

12.49.2.6 item_size()

`size_t ByteToShort::item_size () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 60 of file byte_to_short.h.

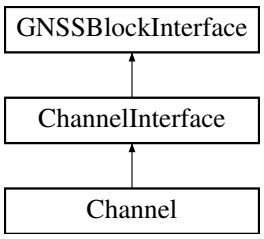
12.49.2.7 role()

`std::string ByteToShort::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 49 of file byte_to_short.h.
The documentation for this class was generated from the following file:

- byte_to_short.h

12.50 Channel Class Reference

This class represents a GNSS channel. It wraps an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction through a Finite State Machine.
`#include <channel.h>`
Inheritance diagram for Channel:



Public Member Functions

- Channel (const ConfigurationInterface *configuration, uint32_t channel, std::shared_ptr< AcquisitionInterface > acq, std::shared_ptr< TrackingInterface > trk, std::shared_ptr< TelemetryDecoderInterface > nav, const std::string &role, const std::string &signal_str, Concurrent_Queue< pmt::pmt_t > *queue)

  Constructor.
- ~Channel ()=default

  Destructor.
- void connect (gr::top_block_sptr top_block) override

  Connects the tracking block to the top_block and to the telemetry.
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_left_block_trk () override

  Gets the GNU Radio tracking block input pointer.

- gr::basic_block_sptr get_right_block_trk () override

  Gets the GNU Radio tracking block output pointer.
- gr::basic_block_sptr get_left_block_acq () override

  Gets the GNU Radio acquisition block input pointer.
- gr::basic_block_sptr get_right_block_acq () override

  Gets the GNU Radio acquisition block output pointer.
- gr::basic_block_sptr get_right_block () override

  Gets the GNU Radio channel block output pointer.
- std::string role () override
- std::string implementation () override

  Returns "Channel".
- size_t item_size () override
- Gnss_Signal get_signal () override
- void start_acquisition () override

  Start the State Machine.
- void stop_channel () override

  Stop the State Machine.
- void set_signal (const Gnss_Signal &gnss_signal_) override

  Sets the channel GNSS signal.
- void assist_acquisition_doppler (double Carrier_Doppler_hz, uint32_t doppler_num_bins=0) override
- std::shared_ptr< AcquisitionInterface > acquisition () const
- std::shared_ptr< TrackingInterface > tracking () const
- std::shared_ptr< TelemetryDecoderInterface > telemetry () const

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.50.1 Detailed Description

This class represents a GNSS channel. It wraps an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction through a Finite State Machine.
Definition at line 60 of file channel.h.

12.50.2 Constructor & Destructor Documentation

12.50.2.1 Channel()

```
Channel::Channel (
            const ConfigurationInterface * configuration,
            uint32_t channel,
            std::shared_ptr< AcquisitionInterface > acq,
            std::shared_ptr< TrackingInterface > trk,
            std::shared_ptr< TelemetryDecoderInterface > nav,
            const std::string & role,
            const std::string & signal_str,
            Concurrent_Queue< pmt::pmt_t > * queue)
```

Constructor.

12.50.2.2 ~Channel()

```
Channel::~Channel ()  [default]
```

Destructor.

12.50.3 Member Function Documentation

12.50.3.1 acquisition()

```
std::shared_ptr< AcquisitionInterface > Channel::acquisition () const  [inline]
```
Definition at line 94 of file channel.h.

12.50.3.2 assist_acquisition_doppler()

```
void Channel::assist_acquisition_doppler (
            double Carrier_Doppler_hz,
            uint32_t doppler_num_bins = 0)  [override], [virtual]
```
Implements ChannelInterface.

12.50.3.3 connect()

```
void Channel::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Connects the tracking block to the top_block and to the telemetry.
Implements GNSSBlockInterface.

12.50.3.4 disconnect()

```
void Channel::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Implements GNSSBlockInterface.

12.50.3.5 get_left_block()

```
gr::basic_block_sptr Channel::get_left_block ()  [override], [virtual]
```
Implements ChannelInterface.

12.50.3.6 get_left_block_acq()

```
gr::basic_block_sptr Channel::get_left_block_acq ()  [override], [virtual]
```
Gets the GNU Radio acquisition block input pointer.
Implements ChannelInterface.

12.50.3.7 get_left_block_trk()

```
gr::basic_block_sptr Channel::get_left_block_trk ()  [override], [virtual]
```
Gets the GNU Radio tracking block input pointer.
Implements ChannelInterface.

12.50.3.8 get_right_block()

```
gr::basic_block_sptr Channel::get_right_block ()  [override], [virtual]
```
Gets the GNU Radio channel block output pointer.
Implements ChannelInterface.

12.50.3.9 get_right_block_acq()

```
gr::basic_block_sptr Channel::get_right_block_acq ()  [override], [virtual]
```
Gets the GNU Radio acquisition block output pointer.
Implements ChannelInterface.

12.50.3.10 get_right_block_trk()

```
gr::basic_block_sptr Channel::get_right_block_trk ()  [override], [virtual]
```
Gets the GNU Radio tracking block output pointer.
Implements ChannelInterface.

12.50.3.11 get_signal()

```
Gnss_Signal Channel::get_signal () [override], [virtual]
```
Implements ChannelInterface.

12.50.3.12 implementation()

```
std::string Channel::implementation () [inline], [override], [virtual]
```
Returns "Channel".
Implements GNSSBlockInterface.
Definition at line 85 of file channel.h.

12.50.3.13 item_size()

```
size_t Channel::item_size () [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 86 of file channel.h.

12.50.3.14 role()

```
std::string Channel::role () [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 84 of file channel.h.

12.50.3.15 set_signal()

```
void Channel::set_signal (
            const Gnss_Signal & gnss_signal_) [override], [virtual]
```
Sets the channel GNSS signal.
Implements ChannelInterface.

12.50.3.16 start_acquisition()

```
void Channel::start_acquisition () [override], [virtual]
```
Start the State Machine.
Implements ChannelInterface.

12.50.3.17 stop_channel()

```
void Channel::stop_channel () [override], [virtual]
```
Stop the State Machine.
Implements ChannelInterface.

12.50.3.18 telemetry()

```
std::shared_ptr< TelemetryDecoderInterface > Channel::telemetry () const [inline]
```
Definition at line 96 of file channel.h.

12.50.3.19 tracking()

```
std::shared_ptr< TrackingInterface > Channel::tracking () const [inline]
```
Definition at line 95 of file channel.h.
The documentation for this class was generated from the following file:

- channel.h

12.51 Channel_Event Class Reference

Public Attributes

- int channel_id
- int event_type

Friends

- channel_event_sptr **channel_event_make** (int channel_id, int event_type)

12.51.1 Detailed Description

Definition at line 34 of file channel_event.h.

12.51.2 Member Data Documentation

12.51.2.1 channel_id

```
int Channel_Event::channel_id
```

Definition at line 37 of file channel_event.h.

12.51.2.2 event_type

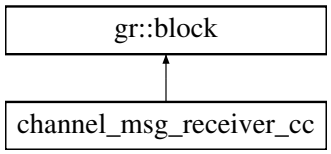
```
int Channel_Event::event_type
```

Definition at line 38 of file channel_event.h.
The documentation for this class was generated from the following file:

- channel_event.h

12.52 channel_msg_receiver_cc Class Reference

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.
`#include <channel_msg_receiver_cc.h>`
Inheritance diagram for channel_msg_receiver_cc:



Public Member Functions

- ~channel_msg_receiver_cc ()=default

  Default destructor.

Friends

- channel_msg_receiver_cc_sptr **channel_msg_receiver_make_cc** (std::shared_ptr< ChannelFsm > channel_fsm, bool repeat)

12.52.1 Detailed Description

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.
Definition at line 40 of file channel_msg_receiver_cc.h.

12.52.2 Constructor & Destructor Documentation

12.52.2.1 ~channel_msg_receiver_cc()

```
channel_msg_receiver_cc::~channel_msg_receiver_cc () [default]
```

Default destructor.

The documentation for this class was generated from the following file:

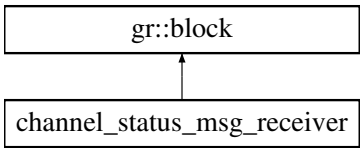
- channel_msg_receiver_cc.h

12.53 channel_status_msg_receiver Class Reference

GNU Radio block that receives asynchronous channel messages from tlm blocks.

`#include <channel_status_msg_receiver.h>`

Inheritance diagram for channel_status_msg_receiver:



Public Member Functions

- ~channel_status_msg_receiver ()=default

  Default destructor.
- std::map< int, std::shared_ptr< Gnss_Synchro > > get_current_status_map ()

  return the current status map of all channels with valid telemetry
- std::map< int, std::shared_ptr< Gnss_Synchro > > get_current_tracking_map ()

  return the current status map of all locked channels, including those without a valid time reference yet (no valid pseudorange). Meant for the secondary-frequency acquisition assistance, which only needs the satellite and its Doppler.
- Monitor_Pvt get_current_status_pvt (double *receiver_time_s=nullptr)

  return the latest successful PVT and optionally elapsed sample time
- void clear_channel_status (int channel_id)

  forget the status of a channel that has been stopped, so that a stale entry is not reported until the channel delivers new observables

Friends

- channel_status_msg_receiver_sptr **channel_status_msg_receiver_make** ()

12.53.1 Detailed Description

GNU Radio block that receives asynchronous channel messages from tlm blocks.

Definition at line 44 of file channel_status_msg_receiver.h.

12.53.2 Constructor & Destructor Documentation

12.53.2.1 ~channel_status_msg_receiver()

```
channel_status_msg_receiver::~channel_status_msg_receiver () [default]
```

Default destructor.

12.53.3 Member Function Documentation

12.53.3.1 clear_channel_status()

```
void channel_status_msg_receiver::clear_channel_status (
            int channel_id)
```

forget the status of a channel that has been stopped, so that a stale entry is not reported until the channel delivers new observables

12.53.3.2 get_current_status_map()

```
std::map< int, std::shared_ptr< Gnss_Synchro > > channel_status_msg_receiver::get_current_↩
status_map ()
```

return the current status map of all channels with valid telemetry

12.53.3.3 get_current_status_pvt()

```
Monitor_Pvt channel_status_msg_receiver::get_current_status_pvt (
            double * receiver_time_s = nullptr)
```

return the latest successful PVT and optionally elapsed sample time

12.53.3.4 get_current_tracking_map()

```
std::map< int, std::shared_ptr< Gnss_Synchro > > channel_status_msg_receiver::get_current_↩
tracking_map ()
```

return the current status map of all locked channels, including those without a valid time reference yet (no valid pseudorange). Meant for the secondary-frequency acquisition assistance, which only needs the satellite and its Doppler.

The documentation for this class was generated from the following file:

- channel_status_msg_receiver.h

12.54 ChannelFsm Class Reference

This class implements a State Machine for channel.

`#include <channel_fsm.h>`

Public Member Functions

- **ChannelFsm** (std::shared_ptr< AcquisitionInterface > acquisition)
- void **set_acquisition** (std::shared_ptr< AcquisitionInterface > acquisition)
- void **set_tracking** (std::shared_ptr< TrackingInterface > tracking)
- void **set_telemetry** (std::shared_ptr< TelemetryDecoderInterface > telemetry)
- void **set_queue** (Concurrent_Queue< pmt::pmt_t > *queue)
- void **set_channel** (uint32_t channel)
- void **start_acquisition** ()
- bool **Event_start_acquisition** ()
- bool **Event_start_acquisition_fpga** ()
- bool **Event_stop_channel** ()
- bool **Event_failed_tracking_standby** ()
- virtual bool **Event_valid_acquisition** ()
- virtual bool **Event_failed_acquisition_repeat** ()
- virtual bool **Event_failed_acquisition_no_repeat** ()

12.54.1 Detailed Description

This class implements a State Machine for channel.

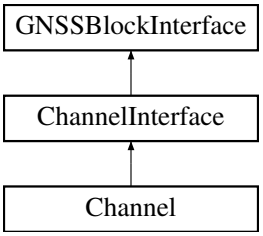
Definition at line 41 of file channel_fsm.h.

The documentation for this class was generated from the following file:

- channel_fsm.h

12.55 ChannelInterface Class Reference

This abstract class represents an interface to a channel GNSS block.
`#include <channel_interface.h>`
Inheritance diagram for ChannelInterface:



Public Member Functions

- virtual gr::basic_block_sptr get_left_block_trk ()=0
- virtual gr::basic_block_sptr get_right_block_trk ()=0
- virtual gr::basic_block_sptr get_left_block_acq ()=0
- virtual gr::basic_block_sptr get_right_block_acq ()=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual Gnss_Signal **get_signal** ()=0
- virtual void start_acquisition ()=0
- virtual void **assist_acquisition_doppler** (double Carrier_Doppler_hz, uint32_t doppler_num_bins=0)=0
- virtual void stop_channel ()=0
- virtual void set_signal (const Gnss_Signal &)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.55.1 Detailed Description

This abstract class represents an interface to a channel GNSS block.
Abstract class for channel blocks. Since all its methods are pure virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 44 of file channel_interface.h.

12.55.2 Member Function Documentation

12.55.2.1 get_left_block()

`virtual gr::basic_block_sptr ChannelInterface::get_left_block () [pure virtual]`
Implements GNSSBlockInterface.

12.55.2.2 get_left_block_acq()

```
virtual gr::basic_block_sptr ChannelInterface::get_left_block_acq ()  [pure virtual]
```
Implemented in Channel.

12.55.2.3 get_left_block_trk()

```
virtual gr::basic_block_sptr ChannelInterface::get_left_block_trk ()  [pure virtual]
```
Implemented in Channel.

12.55.2.4 get_right_block()

```
virtual gr::basic_block_sptr ChannelInterface::get_right_block ()  [pure virtual]
```
Implements GNSSBlockInterface.
Implemented in Channel.

12.55.2.5 get_right_block_acq()

```
virtual gr::basic_block_sptr ChannelInterface::get_right_block_acq ()  [pure virtual]
```
Implemented in Channel.

12.55.2.6 get_right_block_trk()

```
virtual gr::basic_block_sptr ChannelInterface::get_right_block_trk ()  [pure virtual]
```
Implemented in Channel.

12.55.2.7 set_signal()

```
virtual void ChannelInterface::set_signal (
            const Gnss_Signal & )  [pure virtual]
```
Implemented in Channel.

12.55.2.8 start_acquisition()

```
virtual void ChannelInterface::start_acquisition ()  [pure virtual]
```
Implemented in Channel.

12.55.2.9 stop_channel()

```
virtual void ChannelInterface::stop_channel ()  [pure virtual]
```
Implemented in Channel.
The documentation for this class was generated from the following file:

- channel_interface.h

12.56 cl_fft_plan Struct Reference

Public Attributes

- cl_context context
- clFFT_Dim3 n
- clFFT_Dimension dim
- clFFT_DataFormat format
- string * kernel_string
- cl_program program
- cl_fft_kernel_info * kernel_info
- int num_kernels
- cl_kernel twist_kernel
- cl_int temp_buffer_needed

- unsigned last_batch_size
- cl_mem tempmemobj
- cl_mem tempmemobj_real
- cl_mem tempmemobj_imag
- unsigned max_localmem_fft_size
- unsigned max_work_item_per_workgroup
- unsigned max_radix
- unsigned min_mem_coalesce_width
- unsigned num_local_mem_banks

12.56.1 Detailed Description

Definition at line 45 of file fft_internal.h.

12.56.2 Member Data Documentation

12.56.2.1 context

```
cl_context cl_fft_plan::context
```

Definition at line 48 of file fft_internal.h.

12.56.2.2 dim

```
clFFT_Dimension cl_fft_plan::dim
```

Definition at line 54 of file fft_internal.h.

12.56.2.3 format

```
clFFT_DataFormat cl_fft_plan::format
```

Definition at line 57 of file fft_internal.h.

12.56.2.4 kernel_info

```
cl_fft_kernel_info* cl_fft_plan::kernel_info
```

Definition at line 68 of file fft_internal.h.

12.56.2.5 kernel_string

```
string* cl_fft_plan::kernel_string
```

Definition at line 61 of file fft_internal.h.

12.56.2.6 last_batch_size

```
unsigned cl_fft_plan::last_batch_size
```

Definition at line 93 of file fft_internal.h.

12.56.2.7 max_localmem_fft_size

```
unsigned cl_fft_plan::max_localmem_fft_size
```

Definition at line 106 of file fft_internal.h.

12.56.2.8 max_radix

```
unsigned cl_fft_plan::max_radix
```

Definition at line 114 of file fft_internal.h.

12.56.2.9 max_work_item_per_workgroup

```
unsigned cl_fft_plan::max_work_item_per_workgroup
```

Definition at line 110 of file fft_internal.h.

12.56.2.10 min_mem_coalesce_width

```
unsigned cl_fft_plan::min_mem_coalesce_width
```
Definition at line 119 of file fft_internal.h.

12.56.2.11 n

```
clFFT_Dim3 cl_fft_plan::n
```
Definition at line 51 of file fft_internal.h.

12.56.2.12 num_kernels

```
int cl_fft_plan::num_kernels
```
Definition at line 71 of file fft_internal.h.

12.56.2.13 num_local_mem_banks

```
unsigned cl_fft_plan::num_local_mem_banks
```
Definition at line 124 of file fft_internal.h.

12.56.2.14 program

```
cl_program cl_fft_plan::program
```
Definition at line 65 of file fft_internal.h.

12.56.2.15 temp_buffer_needed

```
cl_int cl_fft_plan::temp_buffer_needed
```
Definition at line 84 of file fft_internal.h.

12.56.2.16 tempmemobj

```
cl_mem cl_fft_plan::tempmemobj
```
Definition at line 96 of file fft_internal.h.

12.56.2.17 tempmemobj_imag

```
cl_mem cl_fft_plan::tempmemobj_imag
```
Definition at line 101 of file fft_internal.h.

12.56.2.18 tempmemobj_real

```
cl_mem cl_fft_plan::tempmemobj_real
```
Definition at line 101 of file fft_internal.h.

12.56.2.19 twist_kernel

```
cl_kernel cl_fft_plan::twist_kernel
```
Definition at line 75 of file fft_internal.h.
The documentation for this struct was generated from the following file:

- fft_internal.h

12.57 clFFT_Complex Struct Reference

Public Attributes

- float real
- float imag

12.57.1 Detailed Description

Definition at line 65 of file clFFT.h.

12.57.2 Member Data Documentation

12.57.2.1 imag

`float clFFT_Complex::imag`
Definition at line 68 of file clFFT.h.

12.57.2.2 real

`float clFFT_Complex::real`
Definition at line 67 of file clFFT.h.
The documentation for this struct was generated from the following file:

- clFFT.h

12.58 clFFT_Dim3 Struct Reference

Public Attributes

- unsigned int x
- unsigned int y
- unsigned int z

12.58.1 Detailed Description

Definition at line 52 of file clFFT.h.

12.58.2 Member Data Documentation

12.58.2.1 x

`unsigned int clFFT_Dim3::x`
Definition at line 54 of file clFFT.h.

12.58.2.2 y

`unsigned int clFFT_Dim3::y`
Definition at line 55 of file clFFT.h.

12.58.2.3 z

`unsigned int clFFT_Dim3::z`
Definition at line 56 of file clFFT.h.
The documentation for this struct was generated from the following file:

- clFFT.h

12.59 clFFT_SplitComplex Struct Reference

Public Attributes

- float * real
- float * imag

12.59.1 Detailed Description

Definition at line 59 of file clFFT.h.

12.59.2 Member Data Documentation

12.59.2.1 imag

```
float* clFFT_SplitComplex::imag
```

Definition at line 62 of file clFFT.h.

12.59.2.2 real

```
float* clFFT_SplitComplex::real
```

Definition at line 61 of file clFFT.h.
The documentation for this struct was generated from the following file:

- clFFT.h

12.60 cnav_msg_decoder_t Struct Reference

```
#include <cnav_msg.h>
```

Public Attributes

- cnav_v27_part_t part1
- cnav_v27_part_t part2

12.60.1 Detailed Description

GPS CNAV message lock and decoder object.
Decoder uses two Viterbi decoder objects to ensure the lock is acquired when the input symbol phase is not known.
Definition at line 95 of file cnav_msg.h.

12.60.2 Member Data Documentation

12.60.2.1 part1

```
cnav_v27_part_t cnav_msg_decoder_t::part1
```

Decoder for odd symbol pairs
Definition at line 97 of file cnav_msg.h.

12.60.2.2 part2

```
cnav_v27_part_t cnav_msg_decoder_t::part2
```

Decoder for even symbol pairs
Definition at line 98 of file cnav_msg.h.
The documentation for this struct was generated from the following file:

- cnav_msg.h

12.61 cnav_msg_t Struct Reference

```
#include <cnav_msg.h>
```

Public Attributes

- uint8_t prn
- uint8_t msg_id
- uint32_t tow
- bool alert
- uint8_t raw_msg [GPS_L2C_V27_DECODE_BITS+GPS_L2C_V27_DELAY_BITS]

12.61.1 Detailed Description

GPS CNAV message container.

See also

cnav_msg_decoder_add_symbol

Definition at line 52 of file cnav_msg.h.

12.61.2 Member Data Documentation

12.61.2.1 alert

`bool cnav_msg_t::alert`
CNAV message alert flag
Definition at line 57 of file cnav_msg.h.

12.61.2.2 msg_id

`uint8_t cnav_msg_t::msg_id`
Message id. 0..31
Definition at line 55 of file cnav_msg.h.

12.61.2.3 prn

`uint8_t cnav_msg_t::prn`
SV PRN. 0..31
Definition at line 54 of file cnav_msg.h.

12.61.2.4 raw_msg

`uint8_t cnav_msg_t::raw_msg[GPS_L2C_V27_DECODE_BITS+GPS_L2C_V27_DELAY_BITS]`
RAW MSG for GNSS-SDR
Definition at line 58 of file cnav_msg.h.

12.61.2.5 tow

`uint32_t cnav_msg_t::tow`
GPS ToW in 6-second units. Multiply to 6 to get seconds.
Definition at line 56 of file cnav_msg.h.
The documentation for this struct was generated from the following file:

- cnav_msg.h

12.62 cnav_v27_part_t Struct Reference

`#include <cnav_msg.h>`

Public Attributes

- v27_t dec
- v27_decision_t decisions [GPS_L2_V27_HISTORY_LENGTH_BITS]
- unsigned char symbols [(GPS_L2C_V27_INIT_BITS+GPS_L2C_V27_DECODE_BITS) ∗2]
- size_t n_symbols
- unsigned char decoded [GPS_L2C_V27_DECODE_BITS+GPS_L2C_V27_DELAY_BITS]
- size_t n_decoded
- bool preamble_seen
- bool invert
- bool message_lock
- bool crc_ok
- size_t n_crc_fail
- bool init

12.62.1 Detailed Description

GPS CNAV decoder component. This component controls symbol decoding string.

See also

cnav_msg_decoder_t

Definition at line 67 of file cnav_msg.h.

12.62.2 Member Data Documentation

12.62.2.1 crc_ok

```
bool cnav_v27_part_t::crc_ok
```

Flag that the last message had good CRC
Definition at line 83 of file cnav_msg.h.

12.62.2.2 dec

```
v27_t cnav_v27_part_t::dec
```

Viterbi block decoder object
Definition at line 69 of file cnav_msg.h.

12.62.2.3 decisions

```
v27_decision_t cnav_v27_part_t::decisions[GPS_L2_V27_HISTORY_LENGTH_BITS]
```

Decision graph
Definition at line 70 of file cnav_msg.h.

12.62.2.4 decoded

```
unsigned char cnav_v27_part_t::decoded[GPS_L2C_V27_DECODE_BITS+GPS_L2C_V27_DELAY_BITS]
```

Decode buffer
Definition at line 75 of file cnav_msg.h.

12.62.2.5 init

```
bool cnav_v27_part_t::init
```

Initial state flag. When true, initial bits do not produce output.
Definition at line 85 of file cnav_msg.h.

12.62.2.6 invert

```
bool cnav_v27_part_t::invert
```

When true, indicates the bits are inverted
Definition at line 80 of file cnav_msg.h.

12.62.2.7 message_lock

`bool cnav_v27_part_t::message_lock`
When true, indicates the message boundary is found.
Definition at line 81 of file cnav_msg.h.

12.62.2.8 n_crc_fail

`size_t cnav_v27_part_t::n_crc_fail`
Counter for CRC failures
Definition at line 84 of file cnav_msg.h.

12.62.2.9 n_decoded

`size_t cnav_v27_part_t::n_decoded`
Number of bits in the decode buffer
Definition at line 77 of file cnav_msg.h.

12.62.2.10 n_symbols

`size_t cnav_v27_part_t::n_symbols`
Count of symbols in the symbol buffer
Definition at line 74 of file cnav_msg.h.

12.62.2.11 preamble_seen

`bool cnav_v27_part_t::preamble_seen`
When true, the decode buffer is aligned on preamble.
Definition at line 78 of file cnav_msg.h.

12.62.2.12 symbols

`unsigned char cnav_v27_part_t::symbols[(GPS_L2C_V27_INIT_BITS+GPS_L2C_V27_DECODE_BITS) *2]`
Symbol buffer
Definition at line 72 of file cnav_msg.h.
The documentation for this struct was generated from the following file:

- cnav_msg.h

12.63 Command_Event Class Reference

Public Attributes

- int command_id
- int event_type

Friends

- command_event_sptr **command_event_make** (int command_id, int event_type)

12.63.1 Detailed Description

Definition at line 34 of file command_event.h.

12.63.2 Member Data Documentation

12.63.2.1 command_id

`int Command_Event::command_id`
Definition at line 37 of file command_event.h.

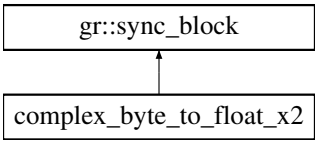
12.63.2.2 event_type

`int Command_Event::event_type`
Definition at line 38 of file command_event.h.
The documentation for this class was generated from the following file:

- command_event.h

12.64 complex_byte_to_float_x2 Class Reference

This class adapts a std::complex<signed char> stream into two 16-bits (short) streams.
`#include <complex_byte_to_float_x2.h>`
Inheritance diagram for complex_byte_to_float_x2:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- complex_byte_to_float_x2_sptr **make_complex_byte_to_float_x2** ()

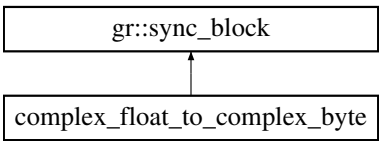
12.64.1 Detailed Description

This class adapts a std::complex<signed char> stream into two 16-bits (short) streams.
Definition at line 41 of file complex_byte_to_float_x2.h.
The documentation for this class was generated from the following file:

- complex_byte_to_float_x2.h

12.65 complex_float_to_complex_byte Class Reference

This class adapts a gr_complex stream into a std::complex<signed char> stream.
`#include <complex_float_to_complex_byte.h>`
Inheritance diagram for complex_float_to_complex_byte:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- complex_float_to_complex_byte_sptr **make_complex_float_to_complex_byte** ()

12.65.1 Detailed Description

This class adapts a gr_complex stream into a std::complex<signed char> stream.
Definition at line 40 of file complex_float_to_complex_byte.h.
The documentation for this class was generated from the following file:

- complex_float_to_complex_byte.h

12.66 Concurrent_Map< Data > Class Template Reference

This class implements a thread-safe std::map.
`#include <concurrent_map.h>`

Public Member Functions

- void write (int key, Data const &data)
- std::map< int, Data > get_map_copy () const &
- std::map< int, Data > get_map_copy () &&
- size_t size () const
- bool read (int key, Data &p_data) const

12.66.1 Detailed Description

template<typename Data>
class Concurrent_Map< Data >

This class implements a thread-safe std::map.
Definition at line 37 of file concurrent_map.h.

12.66.2 Member Function Documentation

12.66.2.1 get_map_copy() [1/2]

```
template<typename Data>
std::map< int, Data > Concurrent_Map< Data >::get_map_copy () &&  [inline]
```

Definition at line 60 of file concurrent_map.h.

12.66.2.2 get_map_copy() [2/2]

```
template<typename Data>
std::map< int, Data > Concurrent_Map< Data >::get_map_copy () const &  [inline]
```

Definition at line 54 of file concurrent_map.h.

12.66.2.3 read()

```
template<typename Data>
bool Concurrent_Map< Data >::read (
            int key,
            Data & p_data) const  [inline]
```

Definition at line 72 of file concurrent_map.h.

12.66.2.4 size()

```
template<typename Data>
size_t Concurrent_Map< Data >::size () const  [inline]
```

Definition at line 66 of file concurrent_map.h.

12.66.2.5 write()

```
template<typename Data>
void Concurrent_Map< Data >::write (
            int key,
            Data const & data)  [inline]
```

Definition at line 40 of file concurrent_map.h.
The documentation for this class was generated from the following file:

- concurrent_map.h

12.67 Concurrent_Queue< Data > Class Template Reference

This class implements a thread-safe std::queue.
`#include <concurrent_queue.h>`

Public Member Functions

- void push (const Data &data)
- void push (Data &&data)
- bool empty () const noexcept
- size_t size () const noexcept
- void clear ()
- bool try_pop (Data &popped_value)
- void wait_and_pop (Data &popped_value)
- bool timed_wait_and_pop (Data &popped_value, int wait_ms)

12.67.1 Detailed Description

template<typename Data>
class Concurrent_Queue< Data >

This class implements a thread-safe std::queue.
Definition at line 38 of file concurrent_queue.h.

12.67.2 Member Function Documentation

12.67.2.1 clear()

```
template<typename Data>
void Concurrent_Queue< Data >::clear ()  [inline]
```

Definition at line 70 of file concurrent_queue.h.

12.67.2.2 empty()

```
template<typename Data>
bool Concurrent_Queue< Data >::empty () const  [inline], [noexcept]
```

Definition at line 59 of file concurrent_queue.h.

12.67.2.3 push() [1/2]

```
template<typename Data>
void Concurrent_Queue< Data >::push (
            const Data & data)  [inline]
```

Definition at line 41 of file concurrent_queue.h.

12.67.2.4 push() [2/2]

```
template<typename Data>
void Concurrent_Queue< Data >::push (
            Data && data)  [inline]
```

Definition at line 50 of file concurrent_queue.h.

12.67.2.5 size()

```
template<typename Data>
size_t Concurrent_Queue< Data >::size () const  [inline], [noexcept]
```

Definition at line 64 of file concurrent_queue.h.

12.67.2.6 timed_wait_and_pop()

```
template<typename Data>
bool Concurrent_Queue< Data >::timed_wait_and_pop (
            Data & popped_value,
            int wait_ms)  [inline]
```

Definition at line 96 of file concurrent_queue.h.

12.67.2.7 try_pop()

```
template<typename Data>
bool Concurrent_Queue< Data >::try_pop (
            Data & popped_value)  [inline]
```

Definition at line 76 of file concurrent_queue.h.

12.67.2.8 wait_and_pop()

```
template<typename Data>
void Concurrent_Queue< Data >::wait_and_pop (
            Data & popped_value)  [inline]
```

Definition at line 88 of file concurrent_queue.h.

The documentation for this class was generated from the following files:

- acquisition_interface.h
- concurrent_queue.h

12.68 HistogramBitSynchronizer::Config Struct Reference

Configuration parameters for HistogramBitSynchronizer.

`#include <bit_synchronizer.h>`

Public Attributes

- int bit_period_ms

  Navigation data bit period in milliseconds.
- int epoch_ms

  Time interval between successive calls to update(), in milliseconds.
- int min_events_for_lock

  Minimum number of detected transition events required before lock evaluation.
- double dominance_ratio

  Required dominance ratio of the winning histogram bin.
- double runner_up_margin

  Required normalized margin between the best and second-best bins.
- int stable_best_required

Required stability of the dominant histogram bin (consecutive evaluations).

- float min_prompt_mag

  Minimum magnitude of the prompt correlator output.

- int transition_window_epochs

  Number of prompt epochs averaged on each side of a candidate edge.

- double transition_confidence

  Minimum normalized transition confidence required for a histogram vote.

- int tentative_events_required

  Number of fresh phase-matching events required after tentative lock.

- bool use_phase_dot_detector

  Select the transition detection method.

12.68.1 Detailed Description

Configuration parameters for HistogramBitSynchronizer.
These parameters define the bit period, the update cadence, the lock criteria, and the transition detection method.
Definition at line 44 of file bit_synchronizer.h.

12.68.2 Constructor & Destructor Documentation

12.68.2.1 Config()

```
HistogramBitSynchronizer::Config::Config () [inline]
```

Definition at line 169 of file bit_synchronizer.h.

12.68.3 Member Data Documentation

12.68.3.1 bit_period_ms

```
int HistogramBitSynchronizer::Config::bit_period_ms
```

Navigation data bit period in milliseconds.
This is the nominal duration of one navigation data bit.
Definition at line 51 of file bit_synchronizer.h.

12.68.3.2 dominance_ratio

```
double HistogramBitSynchronizer::Config::dominance_ratio
```

Required dominance ratio of the winning histogram bin.
Lock requires the most frequent histogram bin to be sufficiently dominant:

$$\text{dominance_ratio} = \frac{\text{best_bin_count}}{\text{total_detected_events}}$$

Guidance:

- Values near 0.5 may lock faster but increase false-lock probability.
- Values closer to 1.0 are conservative and require a clearly dominant phase.

Definition at line 85 of file bit_synchronizer.h.

12.68.3.3 epoch_ms

```
int HistogramBitSynchronizer::Config::epoch_ms
```

Time interval between successive calls to update(), in milliseconds.
This should match the minimum integration interval (epoch) produced by the tracking loop and used to generate the provided prompt correlator output.
Definition at line 59 of file bit_synchronizer.h.

12.68.3.4 min_events_for_lock

`int HistogramBitSynchronizer::Config::min_events_for_lock`
Minimum number of detected transition events required before lock evaluation.
The histogram is built from detected candidate transitions. Lock decisions are not attempted until at least this many events have been accumulated.
Trade-offs:

- Larger values increase robustness against false locks but increase time-to-lock.
- Smaller values reduce time-to-lock but increase sensitivity to noise/spurious transitions.

Definition at line 71 of file bit_synchronizer.h.

12.68.3.5 min_prompt_mag

`float HistogramBitSynchronizer::Config::min_prompt_mag`
Minimum magnitude of the prompt correlator output.
Candidate transition detection is suppressed when $|P| <$ min_prompt_mag, where P is the prompt correlator output.
Use this to avoid counting unreliable transitions when tracking quality is poor or the prompt output is dominated by noise.
Definition at line 116 of file bit_synchronizer.h.

12.68.3.6 runner_up_margin

`double HistogramBitSynchronizer::Config::runner_up_margin`
Required normalized margin between the best and second-best bins.
The margin is computed as (best_count - second_best_count) / total_events. This prevents lock when two candidate phases have similar support.
Definition at line 93 of file bit_synchronizer.h.

12.68.3.7 stable_best_required

`int HistogramBitSynchronizer::Config::stable_best_required`
Required stability of the dominant histogram bin (consecutive evaluations).
Even if the dominance ratio is met, the algorithm requires that the same histogram bin remains dominant for this many consecutive event-driven lock evaluations before declaring lock. Evaluations without a newly detected transition do not count toward this requirement.
This helps prevent locking on transient peaks caused by noise or short-lived disturbances.
Definition at line 105 of file bit_synchronizer.h.

12.68.3.8 tentative_events_required

`int HistogramBitSynchronizer::Config::tentative_events_required`
Number of fresh phase-matching events required after tentative lock.
Once the histogram satisfies its dominance, runner-up margin, and stability requirements, this many subsequent transition events must occur in the same phase bin before lock is declared. Winner stability and tentative validation advance concurrently; they are not applied as serial waiting periods.
Definition at line 144 of file bit_synchronizer.h.

12.68.3.9 transition_confidence

`double HistogramBitSynchronizer::Config::transition_confidence`
Minimum normalized transition confidence required for a histogram vote.
Confidence combines the phase inversion between the pre-edge and post-edge prompt averages with their internal coherence. It lies in the range [-1, 1], with values near 1 indicating a clean polarity inversion.
Definition at line 134 of file bit_synchronizer.h.

12.68.3.10 transition_window_epochs

`int HistogramBitSynchronizer::Config::transition_window_epochs`
Number of prompt epochs averaged on each side of a candidate edge.
A transition is evaluated after this many prompt epochs have been observed on both sides of the candidate boundary. Values larger than one improve rejection of isolated noisy prompt inversions at the cost of a short detection delay.
Definition at line 125 of file bit_synchronizer.h.

12.68.3.11 use_phase_dot_detector

`bool HistogramBitSynchronizer::Config::use_phase_dot_detector`
Select the transition detection method.
If true (recommended), compares coherent prompt averages immediately before and after each candidate boundary. A transition is detected when:

$$-\frac{\Re\{A_{after}A^*_{before}\}}{|A_{after}||A_{before}|}$$

combined with the internal coherence of each averaging window exceeds Config::transition_confidence.
This method is insensitive to constant carrier phase rotations and rejects isolated prompt inversions and ambiguous samples near a candidate boundary.
If false, uses a simpler sign-change detector on the real part:

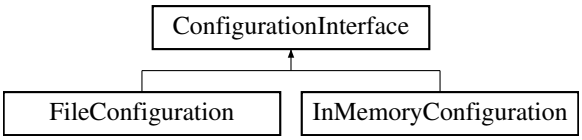
- A candidate transition is detected when sign(Re(P_k)) != sign(Re(P_{k-1})).

This assumes the prompt output is already aligned with the data bit polarity (i.e., stable PLL lock and correct navigation bit polarity mapping).
Definition at line 167 of file bit_synchronizer.h.
The documentation for this struct was generated from the following file:

- bit_synchronizer.h

12.69 ConfigurationInterface Class Reference

This abstract class represents an interface to configuration parameters.
`#include <configuration_interface.h>`
Inheritance diagram for ConfigurationInterface:



Public Member Functions

- virtual std::string **property** (std::string property_name, std::string default_value) const =0
- virtual bool **property** (std::string property_name, bool default_value) const =0
- virtual int64_t **property** (std::string property_name, int64_t default_value) const =0
- virtual uint64_t **property** (std::string property_name, uint64_t default_value) const =0
- virtual int32_t **property** (std::string property_name, int32_t default_value) const =0
- virtual uint32_t **property** (std::string property_name, uint32_t default_value) const =0
- virtual int16_t **property** (std::string property_name, int16_t default_value) const =0
- virtual uint16_t **property** (std::string property_name, uint16_t default_value) const =0
- virtual float **property** (std::string property_name, float default_value) const =0
- virtual double **property** (std::string property_name, double default_value) const =0
- virtual void **set_property** (std::string property_name, std::string value)=0
- virtual bool **is_present** (const std::string &property_name) const =0

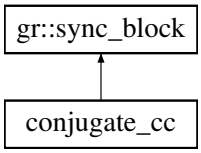
12.69.1 Detailed Description

This abstract class represents an interface to configuration parameters.
The interface defines an accessor method that gets a parameter name as input and returns the value of this parameter, a string, as output. Property names are defined here. This is an abstract class for interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 44 of file configuration_interface.h.
The documentation for this class was generated from the following file:

- configuration_interface.h

12.70 conjugate_cc Class Reference

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
`#include <conjugate_cc.h>`
Inheritance diagram for conjugate_cc:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- conjugate_cc_sptr **make_conjugate_cc** ()

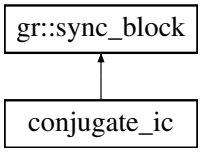
12.70.1 Detailed Description

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
Definition at line 41 of file conjugate_cc.h.
The documentation for this class was generated from the following file:

- conjugate_cc.h

12.71 conjugate_ic Class Reference

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
`#include <conjugate_ic.h>`
Inheritance diagram for conjugate_ic:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- conjugate_ic_sptr **make_conjugate_ic** ()

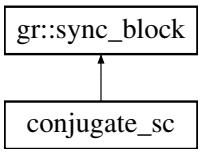
12.71.1 Detailed Description

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
Definition at line 41 of file conjugate_ic.h.
The documentation for this class was generated from the following file:

- conjugate_ic.h

12.72 conjugate_sc Class Reference

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
`#include <conjugate_sc.h>`
Inheritance diagram for conjugate_sc:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- conjugate_sc_sptr **make_conjugate_sc** ()

12.72.1 Detailed Description

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
Definition at line 41 of file conjugate_sc.h.
The documentation for this class was generated from the following file:

- conjugate_sc.h

12.73 ControlThread Class Reference

This class represents the main thread of the application, so the name is ControlThread. This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions.
`#include <control_thread.h>`

Public Member Functions

- ControlThread ()

  Default constructor.
- ControlThread (std::shared_ptr< ConfigurationInterface > configuration)

  Constructor that initializes the class with parameters.
- ~ControlThread ()

  Destructor.
- int run ()

Runs the control thread.

- void set_control_queue (std::shared_ptr< Concurrent_Queue< pmt::pmt_t > > control_queue)

  Sets the control_queue.
- unsigned int processed_control_messages () const
- unsigned int applied_actions () const
- std::shared_ptr< GNSSFlowgraph > flowgraph ()

  Instantiates a flowgraph.

Static Public Attributes

- static ControlThread * me

12.73.1 Detailed Description

This class represents the main thread of the application, so the name is ControlThread. This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions.
Definition at line 65 of file control_thread.h.

12.73.2 Constructor & Destructor Documentation

12.73.2.1 ControlThread() [1/2]

```
ControlThread::ControlThread ()
```

Default constructor.

12.73.2.2 ControlThread() [2/2]

```
ControlThread::ControlThread (
            std::shared_ptr< ConfigurationInterface > configuration)  [explicit]
```

Constructor that initializes the class with parameters.

Parameters

in	*configuration*	Pointer to a ConfigurationInterface

12.73.2.3 ~ControlThread()

```
ControlThread::~ControlThread ()
```

Destructor.

12.73.3 Member Function Documentation

12.73.3.1 applied_actions()

```
unsigned int ControlThread::applied_actions () const  [inline]
```

Definition at line 112 of file control_thread.h.

12.73.3.2 flowgraph()

```
std::shared_ptr< GNSSFlowgraph > ControlThread::flowgraph ()  [inline]
```

Instantiates a flowgraph.

Returns

Returns a smart pointer to a flowgraph object

Definition at line 122 of file control_thread.h.

12.73.3.3 processed_control_messages()

```
unsigned int ControlThread::processed_control_messages () const  [inline]
```
Definition at line 107 of file control_thread.h.

12.73.3.4 run()

```
int ControlThread::run ()
```
Runs the control thread.
This is the main loop that reads and process the control messages:

- Connect the GNSS receiver flowgraph;
- Start the GNSS receiver flowgraph;

while (flowgraph_->running() && !stop_){

- Read control messages and process them; }

12.73.3.5 set_control_queue()

```
void ControlThread::set_control_queue (
            std::shared_ptr< Concurrent_Queue< pmt::pmt_t > > control_queue)
```
Sets the control_queue.

Parameters

in	*std::shared_ptr<Concurrent_Queue<pmt::pmt_t>>*	control_queue

12.73.4 Member Data Documentation

12.73.4.1 me

```
ControlThread* ControlThread::me  [static]
```
Definition at line 68 of file control_thread.h.
The documentation for this class was generated from the following file:

- control_thread.h

12.74 Cpu_Multicorrelator Class Reference

Class that implements carrier wipe-off and correlators.
`#include <cpu_multicorrelator.h>`

Public Member Functions

- bool **init** (int max_signal_length_samples, int n_correlators)
- bool **set_local_code_and_taps** (int code_length_chips, const std::complex< float > *local_code_in, float *shifts_chips)
- bool **set_input_output_vectors** (std::complex< float > *corr_out, const std::complex< float > *sig_in)
- void **update_local_code** (int correlator_length_samples, float rem_code_phase_chips, float code_phase_↩ step_chips)
- bool **Carrier_wipeoff_multicorrelator_resampler** (float rem_carrier_phase_in_rad, float phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, int signal_length_samples)
- bool **free** ()

12.74.1 Detailed Description

Class that implements carrier wipe-off and correlators.
Definition at line 37 of file cpu_multicorrelator.h.
The documentation for this class was generated from the following file:

- cpu_multicorrelator.h

12.75 Cpu_Multicorrelator_16sc Class Reference

Class that implements carrier wipe-off and correlators.
`#include <cpu_multicorrelator_16sc.h>`

Public Member Functions

- bool **init** (int max_signal_length_samples, int n_correlators)
- bool **set_local_code_and_taps** (int code_length_chips, const lv_16sc_t *local_code_in, float *shifts_chips)
- bool **set_input_output_vectors** (lv_16sc_t *corr_out, const lv_16sc_t *sig_in)
- void **update_local_code** (int correlator_length_samples, float rem_code_phase_chips, float code_phase_step_chips)
- bool **Carrier_wipeoff_multicorrelator_resampler** (float rem_carrier_phase_in_rad, float phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, int signal_length_samples)
- bool **free** ()

12.75.1 Detailed Description

Class that implements carrier wipe-off and correlators.
Definition at line 35 of file cpu_multicorrelator_16sc.h.
The documentation for this class was generated from the following file:

- cpu_multicorrelator_16sc.h

12.76 Cpu_Multicorrelator_Real_Codes Class Reference

Class that implements carrier wipe-off and correlators.
`#include <cpu_multicorrelator_real_codes.h>`

Public Member Functions

- void **set_high_dynamics_resampler** (bool use_high_dynamics_resampler)
- bool **init** (int max_signal_length_samples, int n_correlators)
- bool **init** (int max_signal_length_samples, int n_correlators, bool has_data_prompt)
- bool **set_local_code_and_taps** (int code_length_chips, const float *local_code_in, float *shifts_chips)
- bool **set_input_output_vectors** (std::complex< float > *corr_out, const std::complex< float > *sig_in)
- bool **set_data_code_and_prompt_tap** (int code_length_chips, const float *data_code_in, float *prompt_shift_chips)
- bool **set_input_output_vectors** (std::complex< float > *corr_out, std::complex< float > *data_prompt_out, const std::complex< float > *sig_in)
- void **update_local_code** (int correlator_length_samples, float rem_code_phase_chips, float code_phase_step_chips, float code_phase_rate_step_chips=0.0)
- bool **Carrier_wipeoff_multicorrelator_resampler** (float rem_carrier_phase_in_rad, float phase_step_rad, float phase_rate_step_rad, float rem_code_phase_chips, float code_phase_step_chips, float code_phase_rate_step_chips, int signal_length_samples)
- bool **Carrier_wipeoff_multicorrelator_resampler** (float rem_carrier_phase_in_rad, float phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, float code_phase_rate_step_chips, int signal_length_samples)

- bool **Carrier_wipeoff_multicorrelator_resampler_with_data_prompt** (float rem_carrier_phase_in_rad, float phase_step_rad, float phase_rate_step_rad, float rem_code_phase_chips, float code_phase_step_↩ chips, float code_phase_rate_step_chips, int signal_length_samples)
- bool **free** ()

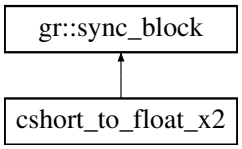
12.76.1 Detailed Description

Class that implements carrier wipe-off and correlators.
Definition at line 38 of file cpu_multicorrelator_real_codes.h.
The documentation for this class was generated from the following file:

- cpu_multicorrelator_real_codes.h

12.77 cshort_to_float_x2 Class Reference

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
`#include <cshort_to_float_x2.h>`
Inheritance diagram for cshort_to_float_x2:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- cshort_to_float_x2_sptr **make_cshort_to_float_x2** ()

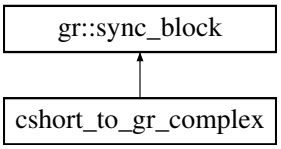
12.77.1 Detailed Description

This class adapts a std::complex<short> stream into two 32-bits (float) streams.
Definition at line 41 of file cshort_to_float_x2.h.
The documentation for this class was generated from the following file:

- cshort_to_float_x2.h

12.78 cshort_to_gr_complex Class Reference

This class adapts a short (16-bits) interleaved sample stream into a std::complex<float> stream.
`#include <cshort_to_gr_complex.h>`
Inheritance diagram for cshort_to_gr_complex:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- cshort_to_gr_complex_sptr **make_cshort_to_gr_complex** ()

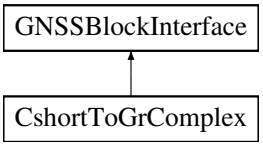
12.78.1 Detailed Description

This class adapts a short (16-bits) interleaved sample stream into a std::complex<float> stream.
Definition at line 40 of file cshort_to_gr_complex.h.
The documentation for this class was generated from the following file:

- cshort_to_gr_complex.h

12.79 CshortToGrComplex Class Reference

Adapts an 16-bits complex sample stream to a float complex stream.
`#include <cshort_to_grcomplex.h>`
Inheritance diagram for CshortToGrComplex:



Public Member Functions

- **CshortToGrComplex** (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Cshort_To_Gr_Complex".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.79.1 Detailed Description

Adapts an 16-bits complex sample stream to a float complex stream.
Definition at line 40 of file cshort_to_grcomplex.h.

12.79.2 Member Function Documentation

12.79.2.1 connect()

```
void CshortToGrComplex::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.79.2.2 disconnect()

```
void CshortToGrComplex::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.79.2.3 get_left_block()

```
gr::basic_block_sptr CshortToGrComplex::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.79.2.4 get_right_block()

```
gr::basic_block_sptr CshortToGrComplex::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.79.2.5 implementation()

```
std::string CshortToGrComplex::implementation ()  [inline], [override], [virtual]
```

Returns "Cshort_To_Gr_Complex".
Implements GNSSBlockInterface.
Definition at line 55 of file cshort_to_grcomplex.h.

12.79.2.6 item_size()

```
size_t CshortToGrComplex::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 60 of file cshort_to_grcomplex.h.

12.79.2.7 role()

```
std::string CshortToGrComplex::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 49 of file cshort_to_grcomplex.h.
The documentation for this class was generated from the following file:

- cshort_to_grcomplex.h

12.80 CubatureFilter Class Reference

Public Member Functions

- **CubatureFilter** (int nx)
- **CubatureFilter** (const arma::vec &x_pred_0, const arma::mat &P_x_pred_0)
- void **initialize** (const arma::mat &x_pred_0, const arma::mat &P_x_pred_0)
- void **predict_sequential** (const arma::vec &x_post, const arma::mat &P_x_post, ModelFunction *transition↩ _fcn, const arma::mat &noise_covariance)
- void **update_sequential** (const arma::vec &z_upd, const arma::vec &x_pred, const arma::mat &P_x_pred, ModelFunction *measurement_fcn, const arma::mat &noise_covariance)
- arma::mat **get_x_pred** () const
- arma::mat **get_P_x_pred** () const
- arma::mat **get_x_est** () const
- arma::mat **get_P_x_est** () const

12.80.1 Detailed Description

Definition at line 54 of file nonlinear_tracking.h.
The documentation for this class was generated from the following file:

- nonlinear_tracking.h

12.81 cuda_multicorrelator Class Reference

Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators.
`#include <cuda_multicorrelator.h>`

Public Member Functions

- bool **init_cuda_integrated_resampler** (int signal_length_samples, int code_length_chips, int n_correlators)
- bool **set_local_code_and_taps** (int code_length_chips, const std::complex< float > *local_codes_in, float *shifts_chips, int n_correlators)
- bool **set_input_output_vectors** (std::complex< float > *corr_out, std::complex< float > *sig_in)
- bool **free_cuda** ()
- bool **Carrier_wipeoff_multicorrelator_resampler_cuda** (float rem_carrier_phase_in_rad, float phase_↩ step_rad, float code_phase_step_chips, float rem_code_phase_chips, int signal_length_samples, int n_↩ correlators)

12.81.1 Detailed Description

Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators.
Definition at line 111 of file cuda_multicorrelator.h.
The documentation for this class was generated from the following file:

- cuda_multicorrelator.h

12.82 CudaPcpsEngine Class Reference

Batched PCPS grid evaluation on a CUDA device.
`#include <cuda_pcps_engine.h>`

Public Types

- enum GridId : int { MAIN_GRID = 0 , STEP2_GRID = 1 , **NUM_GRIDS** = 2 }

  Identifier of the Doppler grid uploaded with set_doppler_wipeoffs().

Public Member Functions

- CudaPcpsEngine (uint32_t fft_size, uint32_t effective_fft_size, uint32_t max_doppler_bins, int device=-1)
- **CudaPcpsEngine** (const CudaPcpsEngine &)=delete
- CudaPcpsEngine & **operator=** (const CudaPcpsEngine &)=delete
- **CudaPcpsEngine** (CudaPcpsEngine &&)=delete
- CudaPcpsEngine & **operator=** (CudaPcpsEngine &&)=delete
- bool is_valid () const

  True when construction succeeded and the engine can be used.
- const std::string & last_error () const

  Human-readable description of the last error (empty if none).
- const std::string & device_name () const

  Name of the CUDA device in use (empty if not valid).
- bool set_fft_codes (const std::complex< float > *fft_codes)

  Upload conj(FFT(local code)), fft_size elements. The upload completes before returning; the caller may reuse fft_↩ codes.
- bool set_doppler_wipeoffs (GridId grid, const std::complex< float > *const *wipeoffs, uint32_t bins)

  Upload the Doppler wipe-off carriers for one grid. Uploads complete before returning; the caller may reuse the host rows.
- bool compute_grid (const std::complex< float > *in, GridId grid, uint32_t bins, uint32_t offset, bool accumulate, float *const *magnitude_out)

  Evaluate the PCPS grid for `bins` Doppler bins.
- size_t device_bytes () const

  Number of bytes allocated on the device by this instance.

12.82.1 Detailed Description

Batched PCPS grid evaluation on a CUDA device.
Typical usage (mirrors pcps_acquisition):

1. Construct with the FFT geometry of the acquisition block.
2. set_doppler_wipeoffs() whenever the Doppler grid changes.
3. set_fft_codes() whenever the local code (PRN) changes.
4. compute_grid() once per dwell; the magnitude grid is written to the host rows supplied by the caller, so the existing CPU peak-search and statistics code can be reused unchanged.

Each instance owns a CUDA stream, so several channels acquiring at the same time run their grids concurrently on the device.
Definition at line 52 of file cuda_pcps_engine.h.

12.82.2 Member Enumeration Documentation

12.82.2.1 GridId

```
enum CudaPcpsEngine::GridId :  int
```

Identifier of the Doppler grid uploaded with set_doppler_wipeoffs().

Enumerator

MAIN_GRID	Coarse grid (doppler_min .. doppler_max, doppler_step).
STEP2_GRID	Fine grid used when make_two_steps=true.

Definition at line 56 of file cuda_pcps_engine.h.

12.82.3 Constructor & Destructor Documentation

12.82.3.1 CudaPcpsEngine()

```
CudaPcpsEngine::CudaPcpsEngine (
            uint32_t fft_size,
            uint32_t effective_fft_size,
            uint32_t max_doppler_bins,
            int device = -1)
```

Parameters

fft_size	Length of the FFTs (input vector length).
effective_fft_size	Number of code-phase cells written per Doppler bin (fft_size, or fft_size/2 when bit_↩ transition_flag is set).
max_doppler_bins	Upper bound on the number of bins in any grid.
device	CUDA device ordinal, or -1 for the current/default device.

References CudaPcpsEngine().
Referenced by CudaPcpsEngine().

12.82.4 Member Function Documentation

12.82.4.1 compute_grid()

```
bool CudaPcpsEngine::compute_grid (
            const std::complex< float > * in,
            GridId grid,
```

```
            uint32_t bins,
            uint32_t offset,
            bool accumulate,
            float *const * magnitude_out)
```

Evaluate the PCPS grid for `bins` Doppler bins.
For every bin k: mag_k[i] (+)= | IFFT(FFT(in .* w_k) .* C)[offset + i] |^2, with C = conj(FFT(code)), i in [0, effective_fft_size).

Parameters

in	Input samples (fft_size complex values, host memory).
grid	Which wipe-off grid to use.
bins	Number of Doppler bins to compute (<= bins uploaded for `grid`).
offset	First sample of the output window (fft_size/2 when bit_transition_flag).
accumulate	false: overwrite the output rows; true: add to their current contents (non-coherent integration across dwells). Accumulation is kept on the device, so the host rows are always the full running sum.
magnitude_out	Array of `bins` host pointers, each with room for effective_fft_size floats.

12.82.4.2 device_bytes()

```
size_t CudaPcpsEngine::device_bytes () const
```

Number of bytes allocated on the device by this instance.

12.82.4.3 device_name()

```
const std::string & CudaPcpsEngine::device_name () const
```

Name of the CUDA device in use (empty if not valid).

12.82.4.4 is_valid()

```
bool CudaPcpsEngine::is_valid () const
```

True when construction succeeded and the engine can be used.

12.82.4.5 last_error()

```
const std::string & CudaPcpsEngine::last_error () const
```

Human-readable description of the last error (empty if none).

12.82.4.6 set_doppler_wipeoffs()

```
bool CudaPcpsEngine::set_doppler_wipeoffs (
            GridId grid,
            const std::complex< float > *const * wipeoffs,
            uint32_t bins)
```

Upload the Doppler wipe-off carriers for one grid. Uploads complete before returning; the caller may reuse the host rows.

Parameters

grid	MAIN_GRID or STEP2_GRID.
wipeoffs	Array of `bins` host pointers, each pointing at fft_size complex samples.
bins	Number of Doppler bins (<= max_doppler_bins).

12.82.4.7 set_fft_codes()

```
bool CudaPcpsEngine::set_fft_codes (
            const std::complex< float > * fft_codes)
```

Upload conj(FFT(local code)), fft_size elements. The upload completes before returning; the caller may reuse fft_codes.

The documentation for this class was generated from the following file:

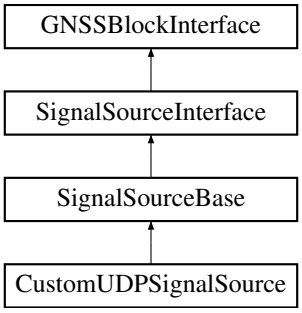
- cuda_pcps_engine.h

12.83 CustomUDPSignalSource Class Reference

This class reads from UDP packets, which streams interleaved I/Q samples over a network.

`#include <custom_udp_signal_source.h>`

Inheritance diagram for CustomUDPSignalSource:



Public Member Functions

- **CustomUDPSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.

- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.83.1 Detailed Description

This class reads from UDP packets, which streams interleaved I/Q samples over a network.
Definition at line 43 of file custom_udp_signal_source.h.

12.83.2 Member Function Documentation

12.83.2.1 connect()

```
void CustomUDPSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.83.2.2 disconnect()

```
void CustomUDPSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.83.2.3 get_left_block()

```
gr::basic_block_sptr CustomUDPSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.83.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr CustomUDPSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.83.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr CustomUDPSignalSource::get_right_block (
            int RF_channel)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.83.2.6 item_size()

```
size_t CustomUDPSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 52 of file custom_udp_signal_source.h.
The documentation for this class was generated from the following file:

- custom_udp_signal_source.h

12.84 Deleter Struct Reference

Public Member Functions

- Deleter (void *buffer)
- template<typename WordType>
  void operator() () const

Public Attributes

- void * buffer_

12.84.1 Detailed Description

Definition at line 75 of file ion_gsms_chunk_data.h.

12.84.2 Constructor & Destructor Documentation

12.84.2.1 Deleter()

```
Deleter::Deleter (
            void * buffer)  [inline], [explicit]
```

Definition at line 79 of file ion_gsms_chunk_data.h.

12.84.3 Member Function Documentation

12.84.3.1 operator()()

```
template<typename WordType>
void Deleter::operator() () const  [inline]
```

Definition at line 83 of file ion_gsms_chunk_data.h.

12.84.4 Member Data Documentation

12.84.4.1 buffer_

```
void* Deleter::buffer_
```

Definition at line 77 of file ion_gsms_chunk_data.h.
The documentation for this struct was generated from the following file:

- ion_gsms_chunk_data.h

12.85 dgps_t Struct Reference

Public Attributes

- gtime_t t0
- double prc
- double rrc
- int iod
- double udre

12.85.1 Detailed Description

Definition at line 681 of file rtklib.h.

12.85.2 Member Data Documentation

12.85.2.1 iod

```
int dgps_t::iod
```

Definition at line 686 of file rtklib.h.

12.85.2.2 prc

```
double dgps_t::prc
```

Definition at line 684 of file rtklib.h.

12.85.2.3 rrc

```
double dgps_t::rrc
```

Definition at line 685 of file rtklib.h.

12.85.2.4 t0

`gtime_t dgps_t::t0`
Definition at line 683 of file rtklib.h.

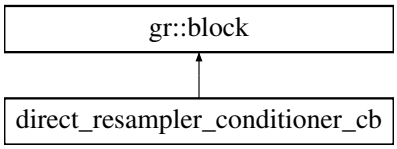
12.85.2.5 udre

`double dgps_t::udre`
Definition at line 687 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.86 direct_resampler_conditioner_cb Class Reference

This class implements a direct resampler conditioner for std::complex<signed char>.
`#include <direct_resampler_conditioner_cb.h>`
Inheritance diagram for direct_resampler_conditioner_cb:



Public Member Functions

- unsigned int sample_freq_in () const
- unsigned int sample_freq_out () const
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- direct_resampler_conditioner_cb_sptr **direct_resampler_make_conditioner_cb** (double sample_freq_in, double sample_freq_out)

12.86.1 Detailed Description

This class implements a direct resampler conditioner for std::complex<signed char>.
Direct resampling without interpolation
Definition at line 46 of file direct_resampler_conditioner_cb.h.

12.86.2 Member Function Documentation

12.86.2.1 sample_freq_in()

`unsigned int direct_resampler_conditioner_cb::sample_freq_in () const [inline]`
Definition at line 51 of file direct_resampler_conditioner_cb.h.

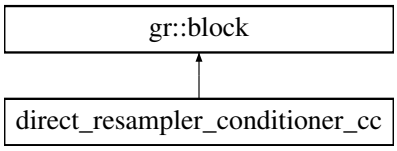
12.86.2.2 sample_freq_out()

`unsigned int direct_resampler_conditioner_cb::sample_freq_out () const [inline]`
Definition at line 56 of file direct_resampler_conditioner_cb.h.
The documentation for this class was generated from the following file:

- direct_resampler_conditioner_cb.h

12.87 direct_resampler_conditioner_cc Class Reference

This class implements a direct resampler conditioner for complex data.
`#include <direct_resampler_conditioner_cc.h>`
Inheritance diagram for direct_resampler_conditioner_cc:



Public Member Functions

- unsigned int sample_freq_in () const
- unsigned int sample_freq_out () const
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- direct_resampler_conditioner_cc_sptr **direct_resampler_make_conditioner_cc** (double sample_freq_in, double sample_freq_out)

12.87.1 Detailed Description

This class implements a direct resampler conditioner for complex data.
Direct resampling without interpolation
Definition at line 51 of file direct_resampler_conditioner_cc.h.

12.87.2 Member Function Documentation

12.87.2.1 sample_freq_in()

`unsigned int direct_resampler_conditioner_cc::sample_freq_in () const [inline]`
Definition at line 55 of file direct_resampler_conditioner_cc.h.

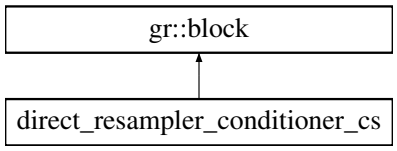
12.87.2.2 sample_freq_out()

`unsigned int direct_resampler_conditioner_cc::sample_freq_out () const [inline]`
Definition at line 60 of file direct_resampler_conditioner_cc.h.
The documentation for this class was generated from the following file:

- direct_resampler_conditioner_cc.h

12.88 direct_resampler_conditioner_cs Class Reference

This class implements a direct resampler conditioner for std::complex<short>.
`#include <direct_resampler_conditioner_cs.h>`
Inheritance diagram for direct_resampler_conditioner_cs:



Public Member Functions

- unsigned int sample_freq_in () const
- unsigned int sample_freq_out () const
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- direct_resampler_conditioner_cs_sptr **direct_resampler_make_conditioner_cs** (double sample_freq_in, double sample_freq_out)

12.88.1 Detailed Description

This class implements a direct resampler conditioner for std::complex<short>.
Direct resampling without interpolation
Definition at line 45 of file direct_resampler_conditioner_cs.h.

12.88.2 Member Function Documentation

12.88.2.1 sample_freq_in()

```
unsigned int direct_resampler_conditioner_cs::sample_freq_in () const  [inline]
```

Definition at line 50 of file direct_resampler_conditioner_cs.h.

12.88.2.2 sample_freq_out()

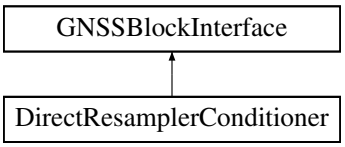
```
unsigned int direct_resampler_conditioner_cs::sample_freq_out () const  [inline]
```

Definition at line 55 of file direct_resampler_conditioner_cs.h.
The documentation for this class was generated from the following file:

- direct_resampler_conditioner_cs.h

12.89 DirectResamplerConditioner Class Reference

Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.
`#include <direct_resampler_conditioner.h>`
Inheritance diagram for DirectResamplerConditioner:



Public Member Functions

- **DirectResamplerConditioner** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream)
- std::string role () override
- std::string implementation () override

  Returns "Direct_Resampler".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.89.1 Detailed Description

Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.
Definition at line 38 of file direct_resampler_conditioner.h.

12.89.2 Member Function Documentation

12.89.2.1 connect()

```
void DirectResamplerConditioner::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.89.2.2 disconnect()

```
void DirectResamplerConditioner::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.89.2.3 get_left_block()

```
gr::basic_block_sptr DirectResamplerConditioner::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.89.2.4 get_right_block()

```
gr::basic_block_sptr DirectResamplerConditioner::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.89.2.5 implementation()

```
std::string DirectResamplerConditioner::implementation ()  [inline], [override], [virtual]
```

Returns "Direct_Resampler".
Implements GNSSBlockInterface.
Definition at line 53 of file direct_resampler_conditioner.h.

12.89.2.6 item_size()

```
size_t DirectResamplerConditioner::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 58 of file direct_resampler_conditioner.h.

12.89.2.7 role()

```
std::string DirectResamplerConditioner::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 47 of file direct_resampler_conditioner.h.
The documentation for this class was generated from the following file:

- direct_resampler_conditioner.h

12.90 Dll_Pll_Conf Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role)

Public Attributes

- std::string item_type {"gr_complex"}
- std::string dump_filename {"./dll_pll_dump.dat"}
- std::string f_error_dump_filename {"./f_error_dump.csv"}
- double fs_in {2000000.0}
- double carrier_lock_th {0.0}
- double bs_dominance_ratio {0.5}
- double bs_runner_up_margin {0.3}
- double bs_transition_confidence {0.6}
- float pll_pull_in_bw_hz {50.0}
- float dll_pull_in_bw_hz {3.0}
- float fll_bw_hz {35.0}
- float pll_bw_hz {35.0}
- float dll_bw_hz {2.0}
- float pll_bw_narrow_hz {5.0}
- float dll_bw_narrow_hz {0.75}
- float early_late_space_chips {0.25}
- float very_early_late_space_chips {0.5}
- float early_late_space_narrow_chips {0.15}
- float very_early_late_space_narrow_chips {0.5}
- float slope {1.0}
- float spc {0.5}
- float y_intercept {1.0}
- float cn0_smoother_alpha {0.002}
- float carrier_lock_test_smoother_alpha {0.002}
- float bs_min_prompt_mag {0.0}
- uint32_t pull_in_time_s {5U}
- uint32_t bit_synchronization_time_limit_s {20U}
- uint32_t vector_length {0U}
- uint32_t smoother_length {10U}
- uint32_t f_error_accumulation {20U}
- uint32_t f_error_step_num {0U}
- double f_error_doppler_step {250.0}
- int32_t fll_filter_order {1}
- int32_t pll_filter_order {3}
- int32_t dll_filter_order {2}
- int32_t extend_correlation_symbols {1}
- int32_t cn0_samples {0}
- int32_t cn0_smoother_samples {200}
- int32_t carrier_lock_test_smoother_samples {25}
- int32_t cn0_min {0}
- int32_t max_code_lock_fail {0}
- int32_t max_carrier_lock_fail {0}
- int32_t bs_stable_best_required {3}
- int32_t bs_min_events_for_lock {6}
- int32_t bs_transition_window_epochs {4}
- int32_t bs_tentative_events_required {2}
- char signal [3] {}
- char system {'G'}

- bool enable_fll_pull_in {false}
- bool enable_fll_steady_state {false}
- bool track_pilot {true}
- bool enable_doppler_correction {false}
- bool carrier_aiding {true}
- bool high_dyn {false}
- bool dump {false}
- bool dump_mat {true}
- bool f_error_dump {false}
- bool tow_to_trk {false}
- bool bs_use_phase_dot_detector {true}
- bool qmboc {true}

  QMBOC local replica (same key as Acquisition_∗.qmboc).
- bool b1c_prompt_use_data_q {true}
- bool b1c_prompt_normalize_power {true}
- float b1c_data_prompt_scale {1.7320508F}
- float b1c_pilot_prompt_scale {1.0F}
- float b1c_secondary_lock_ratio {0.88F}

12.90.1 Detailed Description

Definition at line 32 of file dll_pll_conf.h.

12.90.2 Member Data Documentation

12.90.2.1 b1c_data_prompt_scale

```
float Dll_Pll_Conf::b1c_data_prompt_scale {1.7320508F}
```

Definition at line 102 of file dll_pll_conf.h.

12.90.2.2 b1c_pilot_prompt_scale

```
float Dll_Pll_Conf::b1c_pilot_prompt_scale {1.0F}
```

Definition at line 103 of file dll_pll_conf.h.

12.90.2.3 b1c_prompt_normalize_power

```
bool Dll_Pll_Conf::b1c_prompt_normalize_power {true}
```

Definition at line 101 of file dll_pll_conf.h.

12.90.2.4 b1c_prompt_use_data_q

```
bool Dll_Pll_Conf::b1c_prompt_use_data_q {true}
```

Definition at line 100 of file dll_pll_conf.h.

12.90.2.5 b1c_secondary_lock_ratio

```
float Dll_Pll_Conf::b1c_secondary_lock_ratio {0.88F}
```

Definition at line 104 of file dll_pll_conf.h.

12.90.2.6 bit_synchronization_time_limit_s

```
uint32_t Dll_Pll_Conf::bit_synchronization_time_limit_s {20U}
```

Definition at line 65 of file dll_pll_conf.h.

12.90.2.7 bs_dominance_ratio

```
double Dll_Pll_Conf::bs_dominance_ratio {0.5}
```

Definition at line 44 of file dll_pll_conf.h.

12.90.2.8 bs_min_events_for_lock

```
int32_t Dll_Pll_Conf::bs_min_events_for_lock {6}
```
Definition at line 82 of file dll_pll_conf.h.

12.90.2.9 bs_min_prompt_mag

```
float Dll_Pll_Conf::bs_min_prompt_mag {0.0}
```
Definition at line 63 of file dll_pll_conf.h.

12.90.2.10 bs_runner_up_margin

```
double Dll_Pll_Conf::bs_runner_up_margin {0.3}
```
Definition at line 45 of file dll_pll_conf.h.

12.90.2.11 bs_stable_best_required

```
int32_t Dll_Pll_Conf::bs_stable_best_required {3}
```
Definition at line 81 of file dll_pll_conf.h.

12.90.2.12 bs_tentative_events_required

```
int32_t Dll_Pll_Conf::bs_tentative_events_required {2}
```
Definition at line 84 of file dll_pll_conf.h.

12.90.2.13 bs_transition_confidence

```
double Dll_Pll_Conf::bs_transition_confidence {0.6}
```
Definition at line 46 of file dll_pll_conf.h.

12.90.2.14 bs_transition_window_epochs

```
int32_t Dll_Pll_Conf::bs_transition_window_epochs {4}
```
Definition at line 83 of file dll_pll_conf.h.

12.90.2.15 bs_use_phase_dot_detector

```
bool Dll_Pll_Conf::bs_use_phase_dot_detector {true}
```
Definition at line 97 of file dll_pll_conf.h.

12.90.2.16 carrier_aiding

```
bool Dll_Pll_Conf::carrier_aiding {true}
```
Definition at line 91 of file dll_pll_conf.h.

12.90.2.17 carrier_lock_test_smoother_alpha

```
float Dll_Pll_Conf::carrier_lock_test_smoother_alpha {0.002}
```
Definition at line 62 of file dll_pll_conf.h.

12.90.2.18 carrier_lock_test_smoother_samples

```
int32_t Dll_Pll_Conf::carrier_lock_test_smoother_samples {25}
```
Definition at line 77 of file dll_pll_conf.h.

12.90.2.19 carrier_lock_th

```
double Dll_Pll_Conf::carrier_lock_th {0.0}
```
Definition at line 43 of file dll_pll_conf.h.

12.90.2.20 cn0_min

```
int32_t Dll_Pll_Conf::cn0_min {0}
```
Definition at line 78 of file dll_pll_conf.h.

12.90.2.21 cn0_samples

```
int32_t Dll_Pll_Conf::cn0_samples {0}
```
Definition at line 75 of file dll_pll_conf.h.

12.90.2.22 cn0_smoother_alpha

```
float Dll_Pll_Conf::cn0_smoother_alpha {0.002}
```
Definition at line 61 of file dll_pll_conf.h.

12.90.2.23 cn0_smoother_samples

```
int32_t Dll_Pll_Conf::cn0_smoother_samples {200}
```
Definition at line 76 of file dll_pll_conf.h.

12.90.2.24 dll_bw_hz

```
float Dll_Pll_Conf::dll_bw_hz {2.0}
```
Definition at line 51 of file dll_pll_conf.h.

12.90.2.25 dll_bw_narrow_hz

```
float Dll_Pll_Conf::dll_bw_narrow_hz {0.75}
```
Definition at line 53 of file dll_pll_conf.h.

12.90.2.26 dll_filter_order

```
int32_t Dll_Pll_Conf::dll_filter_order {2}
```
Definition at line 73 of file dll_pll_conf.h.

12.90.2.27 dll_pull_in_bw_hz

```
float Dll_Pll_Conf::dll_pull_in_bw_hz {3.0}
```
Definition at line 48 of file dll_pll_conf.h.

12.90.2.28 dump

```
bool Dll_Pll_Conf::dump {false}
```
Definition at line 93 of file dll_pll_conf.h.

12.90.2.29 dump_filename

```
std::string Dll_Pll_Conf::dump_filename {"./dll_pll_dump.dat"}
```
Definition at line 40 of file dll_pll_conf.h.

12.90.2.30 dump_mat

```
bool Dll_Pll_Conf::dump_mat {true}
```
Definition at line 94 of file dll_pll_conf.h.

12.90.2.31 early_late_space_chips

```
float Dll_Pll_Conf::early_late_space_chips {0.25}
```
Definition at line 54 of file dll_pll_conf.h.

12.90.2.32 early_late_space_narrow_chips

```
float Dll_Pll_Conf::early_late_space_narrow_chips {0.15}
```
Definition at line 56 of file dll_pll_conf.h.

12.90.2.33 enable_doppler_correction

```
bool Dll_Pll_Conf::enable_doppler_correction {false}
```
Definition at line 90 of file dll_pll_conf.h.

12.90.2.34 enable_fll_pull_in

```
bool Dll_Pll_Conf::enable_fll_pull_in {false}
```
Definition at line 87 of file dll_pll_conf.h.

12.90.2.35 enable_fll_steady_state

```
bool Dll_Pll_Conf::enable_fll_steady_state {false}
```
Definition at line 88 of file dll_pll_conf.h.

12.90.2.36 extend_correlation_symbols

```
int32_t Dll_Pll_Conf::extend_correlation_symbols {1}
```
Definition at line 74 of file dll_pll_conf.h.

12.90.2.37 f_error_accumulation

```
uint32_t Dll_Pll_Conf::f_error_accumulation {20U}
```
Definition at line 68 of file dll_pll_conf.h.

12.90.2.38 f_error_doppler_step

```
double Dll_Pll_Conf::f_error_doppler_step {250.0}
```
Definition at line 70 of file dll_pll_conf.h.

12.90.2.39 f_error_dump

```
bool Dll_Pll_Conf::f_error_dump {false}
```
Definition at line 95 of file dll_pll_conf.h.

12.90.2.40 f_error_dump_filename

```
std::string Dll_Pll_Conf::f_error_dump_filename {"./f_error_dump.csv"}
```
Definition at line 41 of file dll_pll_conf.h.

12.90.2.41 f_error_step_num

```
uint32_t Dll_Pll_Conf::f_error_step_num {0U}
```
Definition at line 69 of file dll_pll_conf.h.

12.90.2.42 fll_bw_hz

```
float Dll_Pll_Conf::fll_bw_hz {35.0}
```
Definition at line 49 of file dll_pll_conf.h.

12.90.2.43 fll_filter_order

```
int32_t Dll_Pll_Conf::fll_filter_order {1}
```
Definition at line 71 of file dll_pll_conf.h.

12.90.2.44 fs_in

```
double Dll_Pll_Conf::fs_in {2000000.0}
```
Definition at line 42 of file dll_pll_conf.h.

12.90.2.45 high_dyn

```
bool Dll_Pll_Conf::high_dyn {false}
```
Definition at line 92 of file dll_pll_conf.h.

12.90.2.46 item_type

```
std::string Dll_Pll_Conf::item_type {"gr_complex"}
```
Definition at line 39 of file dll_pll_conf.h.

12.90.2.47 max_carrier_lock_fail

```
int32_t Dll_Pll_Conf::max_carrier_lock_fail {0}
```
Definition at line 80 of file dll_pll_conf.h.

12.90.2.48 max_code_lock_fail

```
int32_t Dll_Pll_Conf::max_code_lock_fail {0}
```
Definition at line 79 of file dll_pll_conf.h.

12.90.2.49 pll_bw_hz

```
float Dll_Pll_Conf::pll_bw_hz {35.0}
```
Definition at line 50 of file dll_pll_conf.h.

12.90.2.50 pll_bw_narrow_hz

```
float Dll_Pll_Conf::pll_bw_narrow_hz {5.0}
```
Definition at line 52 of file dll_pll_conf.h.

12.90.2.51 pll_filter_order

```
int32_t Dll_Pll_Conf::pll_filter_order {3}
```
Definition at line 72 of file dll_pll_conf.h.

12.90.2.52 pll_pull_in_bw_hz

```
float Dll_Pll_Conf::pll_pull_in_bw_hz {50.0}
```
Definition at line 47 of file dll_pll_conf.h.

12.90.2.53 pull_in_time_s

```
uint32_t Dll_Pll_Conf::pull_in_time_s {5U}
```
Definition at line 64 of file dll_pll_conf.h.

12.90.2.54 qmboc

```
bool Dll_Pll_Conf::qmboc {true}
```
QMBOC local replica (same key as Acquisition_*.qmboc).
Definition at line 99 of file dll_pll_conf.h.

12.90.2.55 signal

```
char Dll_Pll_Conf::signal[3] {}
```
Definition at line 85 of file dll_pll_conf.h.

12.90.2.56 slope

```
float Dll_Pll_Conf::slope {1.0}
```
Definition at line 58 of file dll_pll_conf.h.

12.90.2.57 smoother_length

```
uint32_t Dll_Pll_Conf::smoother_length {10U}
```
Definition at line 67 of file dll_pll_conf.h.

12.90.2.58 spc

```
float Dll_Pll_Conf::spc {0.5}
```
Definition at line 59 of file dll_pll_conf.h.

12.90.2.59 system

```
char Dll_Pll_Conf::system {'G'}
```
Definition at line 86 of file dll_pll_conf.h.

12.90.2.60 tow_to_trk

```
bool Dll_Pll_Conf::tow_to_trk {false}
```
Definition at line 96 of file dll_pll_conf.h.

12.90.2.61 track_pilot

```
bool Dll_Pll_Conf::track_pilot {true}
```
Definition at line 89 of file dll_pll_conf.h.

12.90.2.62 vector_length

```
uint32_t Dll_Pll_Conf::vector_length {0U}
```
Definition at line 66 of file dll_pll_conf.h.

12.90.2.63 very_early_late_space_chips

```
float Dll_Pll_Conf::very_early_late_space_chips {0.5}
```
Definition at line 55 of file dll_pll_conf.h.

12.90.2.64 very_early_late_space_narrow_chips

```
float Dll_Pll_Conf::very_early_late_space_narrow_chips {0.5}
```
Definition at line 57 of file dll_pll_conf.h.

12.90.2.65 y_intercept

```
float Dll_Pll_Conf::y_intercept {1.0}
```
Definition at line 60 of file dll_pll_conf.h.
The documentation for this class was generated from the following file:

- dll_pll_conf.h

12.91 Dll_Pll_Conf_Fpga Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role)

Public Attributes

- std::string device_name {"/dev/uio"}
- std::string dump_filename {"./dll_pll_dump.dat"}
- double fs_in {12500000.0}
- double carrier_lock_th {0.0}
- double bs_dominance_ratio {0.5}
- double bs_runner_up_margin {0.3}
- double bs_transition_confidence {0.6}
- float pll_pull_in_bw_hz {50.0}
- float dll_pull_in_bw_hz {3.0}
- float fll_bw_hz {35.0}
- float pll_bw_hz {5.0}
- float dll_bw_hz {0.5}
- float pll_bw_narrow_hz {2.0}
- float dll_bw_narrow_hz {0.25}
- float early_late_space_chips {0.25}
- float very_early_late_space_chips {0.5}
- float early_late_space_narrow_chips {0.15}
- float very_early_late_space_narrow_chips {0.5}
- float slope {1.0}
- float spc {0.5}
- float y_intercept {1.0}
- float cn0_smoother_alpha {0.002}
- float carrier_lock_test_smoother_alpha {0.002}
- float bs_min_prompt_mag {0.0}
- uint32_t pull_in_time_s {5U}
- uint32_t bit_synchronization_time_limit_s {70U}
- uint32_t vector_length {0U}
- uint32_t smoother_length {10U}
- uint32_t code_length_chips {0U}
- uint32_t code_samples_per_chip {0U}
- uint32_t extend_fpga_integration_periods {1}
- uint32_t fpga_integration_period {0}
- int32_t fll_filter_order {1}
- int32_t pll_filter_order {3}
- int32_t dll_filter_order {2}
- int32_t extend_correlation_symbols {1}
- int32_t cn0_samples {0}
- int32_t cn0_min {0}
- int32_t max_code_lock_fail {0}
- int32_t max_carrier_lock_fail {0}
- int32_t cn0_smoother_samples {200}
- int32_t carrier_lock_test_smoother_samples {25}
- int32_t bs_stable_best_required {3}
- int32_t bs_min_events_for_lock {6}
- int32_t bs_transition_window_epochs {4}
- int32_t bs_tentative_events_required {2}
- int32_t * ca_codes {nullptr}
- int32_t * data_codes {nullptr}
- char signal [3] {}
- char system {'G'}
- bool extended_correlation_in_fpga {false}
- bool track_pilot {true}
- bool enable_doppler_correction {false}

- bool enable_fll_pull_in {false}
- bool enable_fll_steady_state {false}
- bool carrier_aiding {true}
- bool high_dyn {false}
- bool dump {false}
- bool dump_mat {true}
- bool bs_use_phase_dot_detector {true}

12.91.1 Detailed Description

Definition at line 34 of file dll_pll_conf_fpga.h.

12.91.2 Member Data Documentation

12.91.2.1 bit_synchronization_time_limit_s

```
uint32_t Dll_Pll_Conf_Fpga::bit_synchronization_time_limit_s {70U}
```
Definition at line 69 of file dll_pll_conf_fpga.h.

12.91.2.2 bs_dominance_ratio

```
double Dll_Pll_Conf_Fpga::bs_dominance_ratio {0.5}
```
Definition at line 46 of file dll_pll_conf_fpga.h.

12.91.2.3 bs_min_events_for_lock

```
int32_t Dll_Pll_Conf_Fpga::bs_min_events_for_lock {6}
```
Definition at line 89 of file dll_pll_conf_fpga.h.

12.91.2.4 bs_min_prompt_mag

```
float Dll_Pll_Conf_Fpga::bs_min_prompt_mag {0.0}
```
Definition at line 66 of file dll_pll_conf_fpga.h.

12.91.2.5 bs_runner_up_margin

```
double Dll_Pll_Conf_Fpga::bs_runner_up_margin {0.3}
```
Definition at line 47 of file dll_pll_conf_fpga.h.

12.91.2.6 bs_stable_best_required

```
int32_t Dll_Pll_Conf_Fpga::bs_stable_best_required {3}
```
Definition at line 88 of file dll_pll_conf_fpga.h.

12.91.2.7 bs_tentative_events_required

```
int32_t Dll_Pll_Conf_Fpga::bs_tentative_events_required {2}
```
Definition at line 91 of file dll_pll_conf_fpga.h.

12.91.2.8 bs_transition_confidence

```
double Dll_Pll_Conf_Fpga::bs_transition_confidence {0.6}
```
Definition at line 48 of file dll_pll_conf_fpga.h.

12.91.2.9 bs_transition_window_epochs

```
int32_t Dll_Pll_Conf_Fpga::bs_transition_window_epochs {4}
```
Definition at line 90 of file dll_pll_conf_fpga.h.

12.91.2.10 bs_use_phase_dot_detector

```
bool Dll_Pll_Conf_Fpga::bs_use_phase_dot_detector {true}
```
Definition at line 108 of file dll_pll_conf_fpga.h.

12.91.2.11 ca_codes

```
int32_t* Dll_Pll_Conf_Fpga::ca_codes {nullptr}
```
Definition at line 93 of file dll_pll_conf_fpga.h.

12.91.2.12 carrier_aiding

```
bool Dll_Pll_Conf_Fpga::carrier_aiding {true}
```
Definition at line 104 of file dll_pll_conf_fpga.h.

12.91.2.13 carrier_lock_test_smoother_alpha

```
float Dll_Pll_Conf_Fpga::carrier_lock_test_smoother_alpha {0.002}
```
Definition at line 65 of file dll_pll_conf_fpga.h.

12.91.2.14 carrier_lock_test_smoother_samples

```
int32_t Dll_Pll_Conf_Fpga::carrier_lock_test_smoother_samples {25}
```
Definition at line 86 of file dll_pll_conf_fpga.h.

12.91.2.15 carrier_lock_th

```
double Dll_Pll_Conf_Fpga::carrier_lock_th {0.0}
```
Definition at line 45 of file dll_pll_conf_fpga.h.

12.91.2.16 cn0_min

```
int32_t Dll_Pll_Conf_Fpga::cn0_min {0}
```
Definition at line 82 of file dll_pll_conf_fpga.h.

12.91.2.17 cn0_samples

```
int32_t Dll_Pll_Conf_Fpga::cn0_samples {0}
```
Definition at line 81 of file dll_pll_conf_fpga.h.

12.91.2.18 cn0_smoother_alpha

```
float Dll_Pll_Conf_Fpga::cn0_smoother_alpha {0.002}
```
Definition at line 64 of file dll_pll_conf_fpga.h.

12.91.2.19 cn0_smoother_samples

```
int32_t Dll_Pll_Conf_Fpga::cn0_smoother_samples {200}
```
Definition at line 85 of file dll_pll_conf_fpga.h.

12.91.2.20 code_length_chips

```
uint32_t Dll_Pll_Conf_Fpga::code_length_chips {0U}
```
Definition at line 72 of file dll_pll_conf_fpga.h.

12.91.2.21 code_samples_per_chip

```
uint32_t Dll_Pll_Conf_Fpga::code_samples_per_chip {0U}
```
Definition at line 73 of file dll_pll_conf_fpga.h.

12.91.2.22 data_codes

```
int32_t* Dll_Pll_Conf_Fpga::data_codes {nullptr}
```
Definition at line 94 of file dll_pll_conf_fpga.h.

12.91.2.23 device_name

```
std::string Dll_Pll_Conf_Fpga::device_name {"/dev/uio"}
```
Definition at line 41 of file dll_pll_conf_fpga.h.

12.91.2.24 dll_bw_hz

```
float Dll_Pll_Conf_Fpga::dll_bw_hz {0.5}
```
Definition at line 54 of file dll_pll_conf_fpga.h.

12.91.2.25 dll_bw_narrow_hz

```
float Dll_Pll_Conf_Fpga::dll_bw_narrow_hz {0.25}
```
Definition at line 56 of file dll_pll_conf_fpga.h.

12.91.2.26 dll_filter_order

```
int32_t Dll_Pll_Conf_Fpga::dll_filter_order {2}
```
Definition at line 79 of file dll_pll_conf_fpga.h.

12.91.2.27 dll_pull_in_bw_hz

```
float Dll_Pll_Conf_Fpga::dll_pull_in_bw_hz {3.0}
```
Definition at line 51 of file dll_pll_conf_fpga.h.

12.91.2.28 dump

```
bool Dll_Pll_Conf_Fpga::dump {false}
```
Definition at line 106 of file dll_pll_conf_fpga.h.

12.91.2.29 dump_filename

```
std::string Dll_Pll_Conf_Fpga::dump_filename {"./dll_pll_dump.dat"}
```
Definition at line 42 of file dll_pll_conf_fpga.h.

12.91.2.30 dump_mat

```
bool Dll_Pll_Conf_Fpga::dump_mat {true}
```
Definition at line 107 of file dll_pll_conf_fpga.h.

12.91.2.31 early_late_space_chips

```
float Dll_Pll_Conf_Fpga::early_late_space_chips {0.25}
```
Definition at line 57 of file dll_pll_conf_fpga.h.

12.91.2.32 early_late_space_narrow_chips

```
float Dll_Pll_Conf_Fpga::early_late_space_narrow_chips {0.15}
```
Definition at line 59 of file dll_pll_conf_fpga.h.

12.91.2.33 enable_doppler_correction

```
bool Dll_Pll_Conf_Fpga::enable_doppler_correction {false}
```
Definition at line 101 of file dll_pll_conf_fpga.h.

12.91.2.34 enable_fll_pull_in

```
bool Dll_Pll_Conf_Fpga::enable_fll_pull_in {false}
```
Definition at line 102 of file dll_pll_conf_fpga.h.

12.91.2.35 enable_fll_steady_state

```
bool Dll_Pll_Conf_Fpga::enable_fll_steady_state {false}
```
Definition at line 103 of file dll_pll_conf_fpga.h.

12.91.2.36 extend_correlation_symbols

```
int32_t Dll_Pll_Conf_Fpga::extend_correlation_symbols {1}
```
Definition at line 80 of file dll_pll_conf_fpga.h.

12.91.2.37 extend_fpga_integration_periods

```
uint32_t Dll_Pll_Conf_Fpga::extend_fpga_integration_periods {1}
```
Definition at line 74 of file dll_pll_conf_fpga.h.

12.91.2.38 extended_correlation_in_fpga

```
bool Dll_Pll_Conf_Fpga::extended_correlation_in_fpga {false}
```
Definition at line 99 of file dll_pll_conf_fpga.h.

12.91.2.39 fll_bw_hz

```
float Dll_Pll_Conf_Fpga::fll_bw_hz {35.0}
```
Definition at line 52 of file dll_pll_conf_fpga.h.

12.91.2.40 fll_filter_order

```
int32_t Dll_Pll_Conf_Fpga::fll_filter_order {1}
```
Definition at line 77 of file dll_pll_conf_fpga.h.

12.91.2.41 fpga_integration_period

```
uint32_t Dll_Pll_Conf_Fpga::fpga_integration_period {0}
```
Definition at line 75 of file dll_pll_conf_fpga.h.

12.91.2.42 fs_in

```
double Dll_Pll_Conf_Fpga::fs_in {12500000.0}
```
Definition at line 44 of file dll_pll_conf_fpga.h.

12.91.2.43 high_dyn

```
bool Dll_Pll_Conf_Fpga::high_dyn {false}
```
Definition at line 105 of file dll_pll_conf_fpga.h.

12.91.2.44 max_carrier_lock_fail

```
int32_t Dll_Pll_Conf_Fpga::max_carrier_lock_fail {0}
```
Definition at line 84 of file dll_pll_conf_fpga.h.

12.91.2.45 max_code_lock_fail

```
int32_t Dll_Pll_Conf_Fpga::max_code_lock_fail {0}
```
Definition at line 83 of file dll_pll_conf_fpga.h.

12.91.2.46 pll_bw_hz

```
float Dll_Pll_Conf_Fpga::pll_bw_hz {5.0}
```
Definition at line 53 of file dll_pll_conf_fpga.h.

12.91.2.47 pll_bw_narrow_hz

```
float Dll_Pll_Conf_Fpga::pll_bw_narrow_hz {2.0}
```
Definition at line 55 of file dll_pll_conf_fpga.h.

12.91.2.48 pll_filter_order

```
int32_t Dll_Pll_Conf_Fpga::pll_filter_order {3}
```
Definition at line 78 of file dll_pll_conf_fpga.h.

12.91.2.49 pll_pull_in_bw_hz

```
float Dll_Pll_Conf_Fpga::pll_pull_in_bw_hz {50.0}
```
Definition at line 50 of file dll_pll_conf_fpga.h.

12.91.2.50 pull_in_time_s

```
uint32_t Dll_Pll_Conf_Fpga::pull_in_time_s {5U}
```
Definition at line 68 of file dll_pll_conf_fpga.h.

12.91.2.51 signal

```
char Dll_Pll_Conf_Fpga::signal[3] {}
```
Definition at line 96 of file dll_pll_conf_fpga.h.

12.91.2.52 slope

```
float Dll_Pll_Conf_Fpga::slope {1.0}
```
Definition at line 61 of file dll_pll_conf_fpga.h.

12.91.2.53 smoother_length

```
uint32_t Dll_Pll_Conf_Fpga::smoother_length {10U}
```
Definition at line 71 of file dll_pll_conf_fpga.h.

12.91.2.54 spc

```
float Dll_Pll_Conf_Fpga::spc {0.5}
```
Definition at line 62 of file dll_pll_conf_fpga.h.

12.91.2.55 system

```
char Dll_Pll_Conf_Fpga::system {'G'}
```
Definition at line 97 of file dll_pll_conf_fpga.h.

12.91.2.56 track_pilot

```
bool Dll_Pll_Conf_Fpga::track_pilot {true}
```
Definition at line 100 of file dll_pll_conf_fpga.h.

12.91.2.57 vector_length

```
uint32_t Dll_Pll_Conf_Fpga::vector_length {0U}
```
Definition at line 70 of file dll_pll_conf_fpga.h.

12.91.2.58 very_early_late_space_chips

```
float Dll_Pll_Conf_Fpga::very_early_late_space_chips {0.5}
```
Definition at line 58 of file dll_pll_conf_fpga.h.

12.91.2.59 very_early_late_space_narrow_chips

```
float Dll_Pll_Conf_Fpga::very_early_late_space_narrow_chips {0.5}
```
Definition at line 60 of file dll_pll_conf_fpga.h.

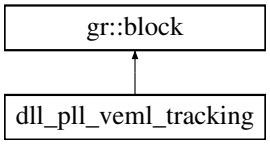
12.91.2.60 y_intercept

```
float Dll_Pll_Conf_Fpga::y_intercept {1.0}
```
Definition at line 63 of file dll_pll_conf_fpga.h.
The documentation for this class was generated from the following file:

- dll_pll_conf_fpga.h

12.92 dll_pll_veml_tracking Class Reference

This class implements a code DLL + carrier PLL tracking block.
`#include <dll_pll_veml_tracking.h>`
Inheritance diagram for dll_pll_veml_tracking:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro *p_gnss_synchro)
- void **start_tracking** ()
- void **stop_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required) override

Static Public Member Functions

- static double f_error_bin_multiplier (uint32_t bin_index)

  Doppler offset multiplier (in units of f_error_doppler_step) for the given frequency-error-reduction bin index: bin 0 -> 0, then alternating outward +1, -1, +2, -2, +3, -3, ...

Friends

- class DllPllTrackingFrequencyErrorTest
- dll_pll_veml_tracking_sptr **dll_pll_veml_make_tracking** (const Dll_Pll_Conf &conf_)

12.92.1 Detailed Description

This class implements a code DLL + carrier PLL tracking block.
Definition at line 62 of file dll_pll_veml_tracking.h.

12.92.2 Member Function Documentation

12.92.2.1 f_error_bin_multiplier()

```
double dll_pll_veml_tracking::f_error_bin_multiplier (
            uint32_t bin_index)  [static]
```

Doppler offset multiplier (in units of f_error_doppler_step) for the given frequency-error-reduction bin index: bin 0 -> 0, then alternating outward +1, -1, +2, -2, +3, -3, ...

12.92.3 Friends And Related Symbol Documentation

12.92.3.1 DllPllTrackingFrequencyErrorTest

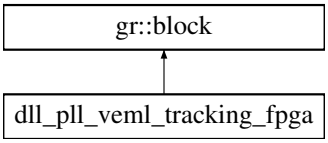
```
friend class DllPllTrackingFrequencyErrorTest  [friend]
```

Definition at line 83 of file dll_pll_veml_tracking.h.
The documentation for this class was generated from the following file:

- dll_pll_veml_tracking.h

12.93 dll_pll_veml_tracking_fpga Class Reference

This class implements a code DLL + carrier PLL tracking block.
`#include <dll_pll_veml_tracking_fpga.h>`
Inheritance diagram for dll_pll_veml_tracking_fpga:



Public Member Functions

- ~dll_pll_veml_tracking_fpga ()

  Destructor.
- void set_channel (uint32_t channel, const std::string &device_io_name)

  Set the channel number and configure some multicorrelator parameters.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro)

  This function is used with two purposes: 1 -> To set the gnss_synchro 2 -> A set_gnss_synchro command with a valid PRN is received when the system is going to run acquisition with that PRN. We can use this command to pre-initialize tracking parameters and variables before the actual acquisition process takes place. In this way we minimize the latency between acquisition and tracking once the acquisition has been made.
- void start_tracking ()

  This function starts the tracking process.
- void stop_tracking ()

  This function sets a flag that makes general_work to stop in order to finish the tracking process.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

  General Work.
- void reset ()

  This function disables the HW multicorrelator in the FPGA in order to stop the tracking process.

Friends

- dll_pll_veml_tracking_fpga_sptr **dll_pll_veml_make_tracking_fpga** (const Dll_Pll_Conf_Fpga &conf_)

12.93.1 Detailed Description

This class implements a code DLL + carrier PLL tracking block.
Definition at line 59 of file dll_pll_veml_tracking_fpga.h.

12.93.2 Constructor & Destructor Documentation

12.93.2.1 ~dll_pll_veml_tracking_fpga()

```
dll_pll_veml_tracking_fpga::~dll_pll_veml_tracking_fpga ()
```

Destructor.

12.93.3 Member Function Documentation

12.93.3.1 general_work()

```
int dll_pll_veml_tracking_fpga::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)
```

General Work.

12.93.3.2 reset()

```
void dll_pll_veml_tracking_fpga::reset ()
```

This function disables the HW multicorrelator in the FPGA in order to stop the tracking process.

12.93.3.3 set_channel()

```
void dll_pll_veml_tracking_fpga::set_channel (
            uint32_t channel,
            const std::string & device_io_name)
```

Set the channel number and configure some multicorrelator parameters.

12.93.3.4 set_gnss_synchro()

```
void dll_pll_veml_tracking_fpga::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)
```

This function is used with two purposes: 1 -> To set the gnss_synchro 2 -> A set_gnss_synchro command with a valid PRN is received when the system is going to run acquisition with that PRN. We can use this command to pre-initialize tracking parameters and variables before the actual acquisition process takes place. In this way we minimize the latency between acquisition and tracking once the acquisition has been made.

12.93.3.5 start_tracking()

```
void dll_pll_veml_tracking_fpga::start_tracking ()
```

This function starts the tracking process.

12.93.3.6 stop_tracking()

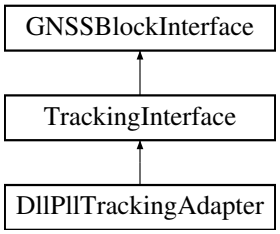
```
void dll_pll_veml_tracking_fpga::stop_tracking ()
```

This function sets a flag that makes general_work to stop in order to finish the tracking process.
The documentation for this class was generated from the following file:

- dll_pll_veml_tracking_fpga.h

12.94 DllPllTrackingAdapter Class Reference

Base class providing shared logic for DLL+PLL tracking loop adapters for GNSS signals.
`#include <dll_pll_tracking_adapter.h>`
Inheritance diagram for DllPllTrackingAdapter:



Public Member Functions

- DllPllTrackingAdapter (const ConfigurationInterface *configuration, std::string role, std::string implementation, unsigned int in_streams, unsigned int out_streams, signal_flag sig_flag)

  Base constructor of a Tracking block adapter.
- ~DllPllTrackingAdapter () override=default

  Default destructor of the Tracking block adapter.
- std::string role () override

  Get role from the Tracking block adapter.
- std::string implementation () override

  Get implementation from the Tracking block adapter.
- size_t item_size () override

  Get item_size from the Tracking block adapter.
- void connect (gr::top_block_sptr top_block) override

  Connect the Tracking block adapter.
- void disconnect (gr::top_block_sptr top_block) override

  Disconnect the sTracking block adapter.
- gr::basic_block_sptr get_left_block () override

  Get left block from the Tracking block adapter.
- gr::basic_block_sptr get_right_block () override

  Get right block from the Tracking block adapter.
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override

  Start the Tracking block.
- void stop_tracking () override

  Stop the Tracking block.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.94.1 Detailed Description

Base class providing shared logic for DLL+PLL tracking loop adapters for GNSS signals.
Definition at line 41 of file dll_pll_tracking_adapter.h.

12.94.2 Constructor & Destructor Documentation

12.94.2.1 DllPllTrackingAdapter()

```
DllPllTrackingAdapter::DllPllTrackingAdapter (
            const ConfigurationInterface * configuration,
            std::string role,
            std::string implementation,
            unsigned int in_streams,
            unsigned int out_streams,
            signal_flag sig_flag)  [explicit]
```

Base constructor of a Tracking block adapter.
References implementation(), and role().

12.94.2.2 ~DllPllTrackingAdapter()

```
DllPllTrackingAdapter::~DllPllTrackingAdapter ()  [override], [default]
```

Default destructor of the Tracking block adapter.

12.94.3 Member Function Documentation

12.94.3.1 connect()

```
void DllPllTrackingAdapter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Connect the Tracking block adapter.
Implements GNSSBlockInterface.

12.94.3.2 disconnect()

```
void DllPllTrackingAdapter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Disconnect the sTracking block adapter.
Implements GNSSBlockInterface.

12.94.3.3 get_left_block()

```
gr::basic_block_sptr DllPllTrackingAdapter::get_left_block ()  [override], [virtual]
```

Get left block from the Tracking block adapter.
Implements GNSSBlockInterface.

12.94.3.4 get_right_block()

```
gr::basic_block_sptr DllPllTrackingAdapter::get_right_block ()  [override], [virtual]
```

Get right block from the Tracking block adapter.
Implements GNSSBlockInterface.

12.94.3.5 implementation()

```
std::string DllPllTrackingAdapter::implementation ()  [inline], [override], [virtual]
```

Get implementation from the Tracking block adapter.
Implements GNSSBlockInterface.
Definition at line 67 of file dll_pll_tracking_adapter.h.
Referenced by DllPllTrackingAdapter().

12.94.3.6 item_size()

```
size_t DllPllTrackingAdapter::item_size ()  [inline], [override], [virtual]
```

Get item_size from the Tracking block adapter.
Implements GNSSBlockInterface.
Definition at line 72 of file dll_pll_tracking_adapter.h.

12.94.3.7 role()

```
std::string DllPllTrackingAdapter::role ()  [inline], [override], [virtual]
```

Get role from the Tracking block adapter.
Implements GNSSBlockInterface.
Definition at line 62 of file dll_pll_tracking_adapter.h.
Referenced by DllPllTrackingAdapter().

12.94.3.8 set_channel()

```
void DllPllTrackingAdapter::set_channel (
            unsigned int channel)  [override], [virtual]
```

Set tracking channel unique ID.
Implements TrackingInterface.

12.94.3.9 set_gnss_synchro()

```
void DllPllTrackingAdapter::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.94.3.10 start_tracking()

```
void DllPllTrackingAdapter::start_tracking ()  [override], [virtual]
```

Start the Tracking block.
Implements TrackingInterface.

12.94.3.11 stop_tracking()

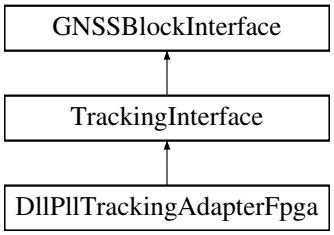
```
void DllPllTrackingAdapter::stop_tracking ()  [override], [virtual]
```

Stop the Tracking block.
Implements TrackingInterface.
The documentation for this class was generated from the following file:

- dll_pll_tracking_adapter.h

12.95 DllPllTrackingAdapterFpga Class Reference

Inheritance diagram for DllPllTrackingAdapterFpga:



Public Member Functions

- **DllPllTrackingAdapterFpga** (const ConfigurationInterface *configuration, const std::string &role, const std::string &implementation, unsigned int in_streams, unsigned int out_streams, signal_flag sig_flag)
- std::string role () override
- std::string implementation () override
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void start_tracking () override
- void stop_tracking () override
- void set_channel (unsigned int channel) override
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.95.1 Detailed Description

Definition at line 33 of file dll_pll_tracking_adapter_fpga.h.

12.95.2 Member Function Documentation

12.95.2.1 connect()

```
void DllPllTrackingAdapterFpga::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.95.2.2 disconnect()

```
void DllPllTrackingAdapterFpga::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.95.2.3 get_left_block()

```
gr::basic_block_sptr DllPllTrackingAdapterFpga::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.95.2.4 get_right_block()

```
gr::basic_block_sptr DllPllTrackingAdapterFpga::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.95.2.5 implementation()

```
std::string DllPllTrackingAdapterFpga::implementation ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 46 of file dll_pll_tracking_adapter_fpga.h.

12.95.2.6 item_size()

```
size_t DllPllTrackingAdapterFpga::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 47 of file dll_pll_tracking_adapter_fpga.h.

12.95.2.7 role()

```
std::string DllPllTrackingAdapterFpga::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 45 of file dll_pll_tracking_adapter_fpga.h.

12.95.2.8 set_channel()

```
void DllPllTrackingAdapterFpga::set_channel (
            unsigned int channel)  [override], [virtual]
```
Implements TrackingInterface.

12.95.2.9 set_gnss_synchro()

```
void DllPllTrackingAdapterFpga::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```
Implements TrackingInterface.

12.95.2.10 start_tracking()

```
void DllPllTrackingAdapterFpga::start_tracking ()  [override], [virtual]
```
Implements TrackingInterface.

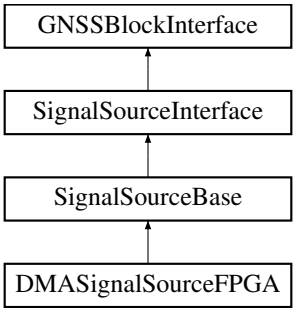
12.95.2.11 stop_tracking()

```
void DllPllTrackingAdapterFpga::stop_tracking ()  [override], [virtual]
```
Implements TrackingInterface.
The documentation for this class was generated from the following file:

- dll_pll_tracking_adapter_fpga.h

12.96 DMASignalSourceFPGA Class Reference

Inheritance diagram for DMASignalSourceFPGA:



Public Member Functions

- **DMASignalSourceFPGA** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- void start () override

  Start the flow of samples if needed.

- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.96.1 Detailed Description

Definition at line 44 of file dma_signal_source_fpga.h.

12.96.2 Member Function Documentation

12.96.2.1 connect()

```
void DMASignalSourceFPGA::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.96.2.2 disconnect()

```
void DMASignalSourceFPGA::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.96.2.3 get_left_block()

```
gr::basic_block_sptr DMASignalSourceFPGA::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.96.2.4 get_right_block()

```
gr::basic_block_sptr DMASignalSourceFPGA::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.96.2.5 item_size()

```
size_t DMASignalSourceFPGA::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 55 of file dma_signal_source_fpga.h.

12.96.2.6 start()

```
void DMASignalSourceFPGA::start ()  [override], [virtual]
```
Start the flow of samples if needed.
Reimplemented from GNSSBlockInterface.
The documentation for this class was generated from the following file:

- dma_signal_source_fpga.h

12.97 DSM_dsm_header Class Reference

Public Attributes

- uint8_t dsm_id {}
- uint8_t dsm_block_id {}

12.97.1 Detailed Description

Definition at line 59 of file osnma_data.h.

12.97.2 Member Data Documentation

12.97.2.1 dsm_block_id

```
uint8_t DSM_dsm_header::dsm_block_id {}
```
Definition at line 64 of file osnma_data.h.

12.97.2.2 dsm_id

```
uint8_t DSM_dsm_header::dsm_id {}
```
Definition at line 63 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.98 DSM_KROOT_message Class Reference

Public Attributes

- std::vector< uint8_t > kroot
- std::vector< uint8_t > ds
- std::vector< uint8_t > p_dk
- uint64_t alpha {}
- uint16_t wn_k {}
- uint8_t nb_dk {}
- uint8_t pkid {}
- uint8_t cidkr {}
- uint8_t reserved1 {}
- uint8_t hf {}
- uint8_t mf {}
- uint8_t ks {}
- uint8_t ts {}

- uint8_t maclt {}
- uint8_t reserved {}
- uint8_t towh_k {}
- bool verified {false}

12.98.1 Detailed Description

Definition at line 116 of file osnma_data.h.

12.98.2 Member Data Documentation

12.98.2.1 alpha

```
uint64_t DSM_KROOT_message::alpha {}
```

Definition at line 124 of file osnma_data.h.

12.98.2.2 cidkr

```
uint8_t DSM_KROOT_message::cidkr {}
```

Definition at line 128 of file osnma_data.h.

12.98.2.3 ds

```
std::vector<uint8_t> DSM_KROOT_message::ds
```

Definition at line 122 of file osnma_data.h.

12.98.2.4 hf

```
uint8_t DSM_KROOT_message::hf {}
```

Definition at line 130 of file osnma_data.h.

12.98.2.5 kroot

```
std::vector<uint8_t> DSM_KROOT_message::kroot
```

Definition at line 121 of file osnma_data.h.

12.98.2.6 ks

```
uint8_t DSM_KROOT_message::ks {}
```

Definition at line 132 of file osnma_data.h.

12.98.2.7 maclt

```
uint8_t DSM_KROOT_message::maclt {}
```

Definition at line 134 of file osnma_data.h.

12.98.2.8 mf

```
uint8_t DSM_KROOT_message::mf {}
```

Definition at line 131 of file osnma_data.h.

12.98.2.9 nb_dk

```
uint8_t DSM_KROOT_message::nb_dk {}
```

Definition at line 126 of file osnma_data.h.

12.98.2.10 p_dk

```
std::vector<uint8_t> DSM_KROOT_message::p_dk
```

Definition at line 123 of file osnma_data.h.

12.98.2.11 pkid

```
uint8_t DSM_KROOT_message::pkid {}
```
Definition at line 127 of file osnma_data.h.

12.98.2.12 reserved

```
uint8_t DSM_KROOT_message::reserved {}
```
Definition at line 135 of file osnma_data.h.

12.98.2.13 reserved1

```
uint8_t DSM_KROOT_message::reserved1 {}
```
Definition at line 129 of file osnma_data.h.

12.98.2.14 towh_k

```
uint8_t DSM_KROOT_message::towh_k {}
```
Definition at line 136 of file osnma_data.h.

12.98.2.15 ts

```
uint8_t DSM_KROOT_message::ts {}
```
Definition at line 133 of file osnma_data.h.

12.98.2.16 verified

```
bool DSM_KROOT_message::verified {false}
```
Definition at line 137 of file osnma_data.h.

12.98.2.17 wn_k

```
uint16_t DSM_KROOT_message::wn_k {}
```
Definition at line 125 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.99 DSM_nma_header Class Reference

Public Attributes

- uint8_t nmas {}
- uint8_t cid {}
- uint8_t cpks {}
- bool reserved {}

12.99.1 Detailed Description

Definition at line 48 of file osnma_data.h.

12.99.2 Member Data Documentation

12.99.2.1 cid

```
uint8_t DSM_nma_header::cid {}
```
Definition at line 53 of file osnma_data.h.

12.99.2.2 cpks

```
uint8_t DSM_nma_header::cpks {}
```
Definition at line 54 of file osnma_data.h.

12.99.2.3 nmas

```
uint8_t DSM_nma_header::nmas {}
```
Definition at line 52 of file osnma_data.h.

12.99.2.4 reserved

```
bool DSM_nma_header::reserved {}
```
Definition at line 55 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.100 DSM_PKR_message Class Reference

Public Attributes

- std::array< uint8_t, 128 > itn {}
- std::vector< uint8_t > npk
- std::vector< uint8_t > p_dp
- uint8_t nb_dp {}
- uint8_t mid {}
- uint8_t npkt {}
- uint8_t npktid {}

12.100.1 Detailed Description

Definition at line 101 of file osnma_data.h.

12.100.2 Member Data Documentation

12.100.2.1 itn

```
std::array<uint8_t, 128> DSM_PKR_message::itn {}
```
Definition at line 106 of file osnma_data.h.

12.100.2.2 mid

```
uint8_t DSM_PKR_message::mid {}
```
Definition at line 110 of file osnma_data.h.

12.100.2.3 nb_dp

```
uint8_t DSM_PKR_message::nb_dp {}
```
Definition at line 109 of file osnma_data.h.

12.100.2.4 npk

```
std::vector<uint8_t> DSM_PKR_message::npk
```
Definition at line 107 of file osnma_data.h.

12.100.2.5 npkt

```
uint8_t DSM_PKR_message::npkt {}
```
Definition at line 111 of file osnma_data.h.

12.100.2.6 npktid

```
uint8_t DSM_PKR_message::npktid {}
```
Definition at line 112 of file osnma_data.h.

12.100.2.7 p_dp

```
std::vector<uint8_t> DSM_PKR_message::p_dp
```
Definition at line 108 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.101 eph_t Struct Reference

Public Attributes

- int sat
- int iode
- int iodc
- int sva
- int svh
- int week
- int code
- int flag
- gtime_t toe
- gtime_t toc
- gtime_t ttr
- double A
- double e
- double i0
- double OMG0
- double omg
- double M0
- double deln
- double OMGd
- double idot
- double crc
- double crs
- double cuc
- double cus
- double cic
- double cis
- double toes
- double fit
- double f0
- double f1
- double f2
- double tgd [4]
- double isc [4]
- double Adot
- double ndot
- float has_clock_correction_m
- float has_orbit_radial_correction_m
- float has_orbit_in_track_correction_m
- float has_orbit_cross_track_correction_m

- bool apply_has_corrections
- int cnav_uraed
- int cnav_uraned0
- int cnav_uraned1
- int cnav_uraned2
- double cnav_top
- int cnav_wnop
- int cnav_ura_valid

12.101.1 Detailed Description

Definition at line 461 of file rtklib.h.

12.101.2 Member Data Documentation

12.101.2.1 A

```
double eph_t::A
```

Definition at line 472 of file rtklib.h.

12.101.2.2 Adot

```
double eph_t::Adot
```

Definition at line 482 of file rtklib.h.

12.101.2.3 apply_has_corrections

```
bool eph_t::apply_has_corrections
```

Definition at line 487 of file rtklib.h.

12.101.2.4 cic

```
double eph_t::cic
```

Definition at line 473 of file rtklib.h.

12.101.2.5 cis

```
double eph_t::cis
```

Definition at line 473 of file rtklib.h.

12.101.2.6 cnav_top

```
double eph_t::cnav_top
```

Definition at line 492 of file rtklib.h.

12.101.2.7 cnav_ura_valid

```
int eph_t::cnav_ura_valid
```

Definition at line 494 of file rtklib.h.

12.101.2.8 cnav_uraed

```
int eph_t::cnav_uraed
```

Definition at line 488 of file rtklib.h.

12.101.2.9 cnav_uraned0

```
int eph_t::cnav_uraned0
```

Definition at line 489 of file rtklib.h.

12.101.2.10 cnav_uraned1

`int eph_t::cnav_uraned1`
Definition at line 490 of file rtklib.h.

12.101.2.11 cnav_uraned2

`int eph_t::cnav_uraned2`
Definition at line 491 of file rtklib.h.

12.101.2.12 cnav_wnop

`int eph_t::cnav_wnop`
Definition at line 493 of file rtklib.h.

12.101.2.13 code

`int eph_t::code`
Definition at line 468 of file rtklib.h.

12.101.2.14 crc

`double eph_t::crc`
Definition at line 473 of file rtklib.h.

12.101.2.15 crs

`double eph_t::crs`
Definition at line 473 of file rtklib.h.

12.101.2.16 cuc

`double eph_t::cuc`
Definition at line 473 of file rtklib.h.

12.101.2.17 cus

`double eph_t::cus`
Definition at line 473 of file rtklib.h.

12.101.2.18 deln

`double eph_t::deln`
Definition at line 472 of file rtklib.h.

12.101.2.19 e

`double eph_t::e`
Definition at line 472 of file rtklib.h.

12.101.2.20 f0

`double eph_t::f0`
Definition at line 476 of file rtklib.h.

12.101.2.21 f1

`double eph_t::f1`
Definition at line 476 of file rtklib.h.

12.101.2.22 f2

`double eph_t::f2`
Definition at line 476 of file rtklib.h.

12.101.2.23 fit

`double eph_t::fit`
Definition at line 475 of file rtklib.h.

12.101.2.24 flag

`int eph_t::flag`
Definition at line 469 of file rtklib.h.

12.101.2.25 has_clock_correction_m

`float eph_t::has_clock_correction_m`
Definition at line 483 of file rtklib.h.

12.101.2.26 has_orbit_cross_track_correction_m

`float eph_t::has_orbit_cross_track_correction_m`
Definition at line 486 of file rtklib.h.

12.101.2.27 has_orbit_in_track_correction_m

`float eph_t::has_orbit_in_track_correction_m`
Definition at line 485 of file rtklib.h.

12.101.2.28 has_orbit_radial_correction_m

`float eph_t::has_orbit_radial_correction_m`
Definition at line 484 of file rtklib.h.

12.101.2.29 i0

`double eph_t::i0`
Definition at line 472 of file rtklib.h.

12.101.2.30 idot

`double eph_t::idot`
Definition at line 472 of file rtklib.h.

12.101.2.31 iodc

`int eph_t::iodc`
Definition at line 464 of file rtklib.h.

12.101.2.32 iode

`int eph_t::iode`
Definition at line 464 of file rtklib.h.

12.101.2.33 isc

`double eph_t::isc[4]`
Definition at line 481 of file rtklib.h.

12.101.2.34 M0

`double eph_t::M0`
Definition at line 472 of file rtklib.h.

12.101.2.35 ndot

`double eph_t::ndot`
Definition at line 482 of file rtklib.h.

12.101.2.36 omg

`double eph_t::omg`
Definition at line 472 of file rtklib.h.

12.101.2.37 OMG0

`double eph_t::OMG0`
Definition at line 472 of file rtklib.h.

12.101.2.38 OMGd

`double eph_t::OMGd`
Definition at line 472 of file rtklib.h.

12.101.2.39 sat

`int eph_t::sat`
Definition at line 463 of file rtklib.h.

12.101.2.40 sva

`int eph_t::sva`
Definition at line 465 of file rtklib.h.

12.101.2.41 svh

`int eph_t::svh`
Definition at line 466 of file rtklib.h.

12.101.2.42 tgd

`double eph_t::tgd[4]`
Definition at line 477 of file rtklib.h.

12.101.2.43 toc

`gtime_t eph_t::toc`
Definition at line 470 of file rtklib.h.

12.101.2.44 toe

`gtime_t eph_t::toe`
Definition at line 470 of file rtklib.h.

12.101.2.45 toes

`double eph_t::toes`
Definition at line 474 of file rtklib.h.

12.101.2.46 ttr

```
gtime_t eph_t::ttr
```
Definition at line 470 of file rtklib.h.

12.101.2.47 week

```
int eph_t::week
```
Definition at line 467 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.102 erp_t Struct Reference

Public Attributes

- int n
- int nmax
- erpd_t * data

12.102.1 Detailed Description

Definition at line 421 of file rtklib.h.

12.102.2 Member Data Documentation

12.102.2.1 data

```
erpd_t* erp_t::data
```
Definition at line 424 of file rtklib.h.

12.102.2.2 n

```
int erp_t::n
```
Definition at line 423 of file rtklib.h.

12.102.2.3 nmax

```
int erp_t::nmax
```
Definition at line 423 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.103 erpd_t Struct Reference

Public Attributes

- double mjd
- double xp
- double yp
- double xpr
- double ypr
- double ut1_utc
- double lod

12.103.1 Detailed Description

Definition at line 411 of file rtklib.h.

12.103.2 Member Data Documentation

12.103.2.1 lod

`double erpd_t::lod`
Definition at line 417 of file rtklib.h.

12.103.2.2 mjd

`double erpd_t::mjd`
Definition at line 413 of file rtklib.h.

12.103.2.3 ut1_utc

`double erpd_t::ut1_utc`
Definition at line 416 of file rtklib.h.

12.103.2.4 xp

`double erpd_t::xp`
Definition at line 414 of file rtklib.h.

12.103.2.5 xpr

`double erpd_t::xpr`
Definition at line 415 of file rtklib.h.

12.103.2.6 yp

`double erpd_t::yp`
Definition at line 414 of file rtklib.h.

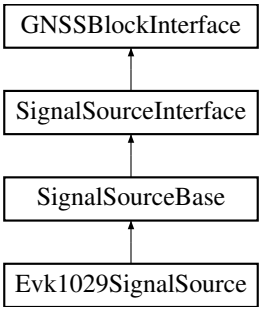
12.103.2.7 ypr

`double erpd_t::ypr`
Definition at line 415 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.104 Evk1029SignalSource Class Reference

This class reads a SAPHYRION EVK1029 raw capture file.
`#include <evk1029_signal_source.h>`
Inheritance diagram for Evk1029SignalSource:



Public Member Functions

- **Evk1029SignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.104.1 Detailed Description

This class reads a SAPHYRION EVK1029 raw capture file.
Definition at line 58 of file evk1029_signal_source.h.

12.104.2 Member Function Documentation

12.104.2.1 connect()

```
void Evk1029SignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.104.2.2 disconnect()

```
void Evk1029SignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.104.2.3 get_right_block()

```
gr::basic_block_sptr Evk1029SignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.104.2.4 item_size()

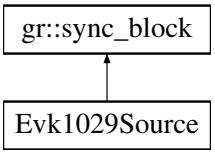
```
size_t Evk1029SignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 70 of file evk1029_signal_source.h.
The documentation for this class was generated from the following file:

- evk1029_signal_source.h

12.105 Evk1029Source Class Reference

Reads a continuous, header-less EVK1029 raw capture file and unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first).
`#include <evk1029_source.h>`
Inheritance diagram for Evk1029Source:



Friends

- Evk1029Source_sptr **evk1029_make_source** (const std::string &filename, Concurrent_Queue< pmt::pmt↩ _t > *queue, double sampling_frequency, uint64_t bytes_to_skip=0)

12.105.1 Detailed Description

Reads a continuous, header-less EVK1029 raw capture file and unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first).
Definition at line 79 of file evk1029_source.h.
The documentation for this class was generated from the following file:

- evk1029_source.h

12.106 Exponential_Smoother Class Reference

Class that implements a first-order exponential smoother.
`#include <exponential_smoother.h>`

Public Member Functions

- Exponential_Smoother ()

  Constructor.
- ~Exponential_Smoother ()=default

  Destructor.
- Exponential_Smoother (Exponential_Smoother &&)=default

  Move operator.
- Exponential_Smoother & operator= (Exponential_Smoother &&)=default

  Move assignment operator.
- void set_alpha (float alpha)

  0 < alpha < 1. The higher, the most responsive, but more variance. Default value: 0.001
- void set_samples_for_initialization (int num_samples)

  Number of samples averaged for initialization. Default value: 200.
- void **reset** ()

- void **set_min_value** (float value)
- void **set_offset** (float offset)
- float **smooth** (float raw)
- double **smooth** (double raw)

12.106.1 Detailed Description

Class that implements a first-order exponential smoother.
smoothed_value[k] = alpha ∗ raw + (1-alpha) ∗ smoothed_value[k-1]
The length of the initialization can be controlled with set_samples_for_initialization(int num_samples)
Definition at line 39 of file exponential_smoother.h.

12.106.2 Constructor & Destructor Documentation

12.106.2.1 Exponential_Smoother() [1/2]

```
Exponential_Smoother::Exponential_Smoother ()
```

Constructor.
Referenced by Exponential_Smoother(), and operator=().

12.106.2.2 ∼Exponential_Smoother()

```
Exponential_Smoother::~Exponential_Smoother ()  [default]
```

Destructor.

12.106.2.3 Exponential_Smoother() [2/2]

```
Exponential_Smoother::Exponential_Smoother (
            Exponential_Smoother && )  [default]
```

Move operator.
References Exponential_Smoother().

12.106.3 Member Function Documentation

12.106.3.1 operator=()

```
Exponential_Smoother & Exponential_Smoother::operator= (
            Exponential_Smoother && )  [default]
```

Move assignment operator.
References Exponential_Smoother().

12.106.3.2 set_alpha()

```
void Exponential_Smoother::set_alpha (
            float alpha)
```

0 < alpha < 1. The higher, the most responsive, but more variance. Default value: 0.001

12.106.3.3 set_samples_for_initialization()

```
void Exponential_Smoother::set_samples_for_initialization (
            int num_samples)
```

Number of samples averaged for initialization. Default value: 200.
The documentation for this class was generated from the following file:

- exponential_smoother.h

12.107 exterr_t Struct Reference

Public Attributes

- int ena [4]
- double cerr [4][NFREQ ∗2]
- double perr [4][NFREQ ∗2]
- double gpsglob [NFREQ]
- double gloicb [NFREQ]

12.107.1 Detailed Description

Definition at line 965 of file rtklib.h.

12.107.2 Member Data Documentation

12.107.2.1 cerr

```
double exterr_t::cerr[4][NFREQ *2]
```

Definition at line 968 of file rtklib.h.

12.107.2.2 ena

```
int exterr_t::ena[4]
```

Definition at line 967 of file rtklib.h.

12.107.2.3 gloicb

```
double exterr_t::gloicb[NFREQ]
```

Definition at line 971 of file rtklib.h.

12.107.2.4 gpsglob

```
double exterr_t::gpsglob[NFREQ]
```

Definition at line 970 of file rtklib.h.

12.107.2.5 perr

```
double exterr_t::perr[4][NFREQ *2]
```

Definition at line 969 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.108 fcbd_t Struct Reference

Public Attributes

- gtime_t ts
- gtime_t te
- double bias [MAXSAT][3]
- double std [MAXSAT][3]

12.108.1 Detailed Description

Definition at line 593 of file rtklib.h.

12.108.2 Member Data Documentation

12.108.2.1 bias

`double fcbd_t::bias[MAXSAT][3]`
Definition at line 596 of file rtklib.h.

12.108.2.2 std

`double fcbd_t::std[MAXSAT][3]`
Definition at line 597 of file rtklib.h.

12.108.2.3 te

`gtime_t fcbd_t::te`
Definition at line 595 of file rtklib.h.

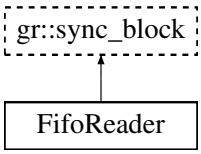
12.108.2.4 ts

`gtime_t fcbd_t::ts`
Definition at line 595 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.109 FifoReader Class Reference

Inheritance diagram for FifoReader:



Public Types

- using sptr = gnss_shared_ptr<FifoReader>
  static function to create a class instance

Public Member Functions

- bool start ()
  initialize istream resource for FIFO
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Static Public Member Functions

- static sptr **make** (const std::string &file_name, const std::string &sample_type)

12.109.1 Detailed Description

Definition at line 31 of file fifo_reader.h.

12.109.2 Member Typedef Documentation

12.109.2.1 sptr

```
using FifoReader::sptr = gnss_shared_ptr<FifoReader>
```

static function to create a class instance
Definition at line 35 of file fifo_reader.h.

12.109.3 Member Function Documentation

12.109.3.1 start()

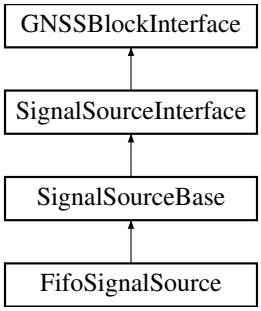
```
bool FifoReader::start ()
```

initialize istream resource for FIFO
The documentation for this class was generated from the following file:

- fifo_reader.h

12.110 FifoSignalSource Class Reference

Class that reads a sample stream from a Unix FIFO.
`#include <fifo_signal_source.h>`
Inheritance diagram for FifoSignalSource:



Public Member Functions

- **FifoSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)
- void connect (gr::top_block_sptr top_block) override

  override methods from GNSSBlockInterface
- void disconnect (gr::top_block_sptr top_block) override
- size_t item_size () override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.110.1 Detailed Description

Class that reads a sample stream from a Unix FIFO.
This class supports the following properties:
.filename - the path to the input file

- may be overridden by the -signal_source or -s command-line arguments

.sample_type - data type read out from the FIFO. default ishort ;

- note: not output format. that is always gr_complex

.dump - whether to archive input data
.dump_filename - if dumping, path to file for output
Definition at line 50 of file fifo_signal_source.h.

12.110.2 Member Function Documentation

12.110.2.1 connect()

```
void FifoSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

override methods from GNSSBlockInterface
Implements GNSSBlockInterface.

12.110.2.2 disconnect()

```
void FifoSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.110.2.3 get_left_block()

```
gr::basic_block_sptr FifoSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.110.2.4 get_right_block()

```
gr::basic_block_sptr FifoSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.110.2.5 item_size()

```
size_t FifoSignalSource::item_size ()  [override], [virtual]
```

Implements GNSSBlockInterface.
The documentation for this class was generated from the following file:

- fifo_signal_source.h

12.111 file_t Struct Reference

Public Attributes

- FILE * fp = nullptr
- FILE * fp_tag = nullptr
- FILE * fp_tmp = nullptr
- FILE * fp_tag_tmp = nullptr
- std::string path
- std::string openpath
- int mode = 0
- int timetag
- int repmode = 0
- int offset = 0
- gtime_t time = {}
- gtime_t wtime = {}
- unsigned int tick = 0
- unsigned int tick_f = 0
- unsigned int fpos = 0
- double start = 0
- double speed = 0
- double swapintv = 0
- lock_t lock

12.111.1 Detailed Description

Definition at line 1183 of file rtklib.h.

12.111.2 Member Data Documentation

12.111.2.1 fp

```
FILE* file_t::fp = nullptr
```

Definition at line 1185 of file rtklib.h.

12.111.2.2 fp_tag

```
FILE* file_t::fp_tag = nullptr
```

Definition at line 1186 of file rtklib.h.

12.111.2.3 fp_tag_tmp

```
FILE* file_t::fp_tag_tmp = nullptr
```

Definition at line 1188 of file rtklib.h.

12.111.2.4 fp_tmp

```
FILE* file_t::fp_tmp = nullptr
```

Definition at line 1187 of file rtklib.h.

12.111.2.5 fpos

```
unsigned int file_t::fpos = 0
```

Definition at line 1199 of file rtklib.h.

12.111.2.6 lock

```
lock_t file_t::lock
```

Definition at line 1203 of file rtklib.h.

12.111.2.7 mode

```
int file_t::mode = 0
```
Definition at line 1191 of file rtklib.h.

12.111.2.8 offset

```
int file_t::offset = 0
```
Definition at line 1194 of file rtklib.h.

12.111.2.9 openpath

```
std::string file_t::openpath
```
Definition at line 1190 of file rtklib.h.

12.111.2.10 path

```
std::string file_t::path
```
Definition at line 1189 of file rtklib.h.

12.111.2.11 repmode

```
int file_t::repmode = 0
```
Definition at line 1193 of file rtklib.h.

12.111.2.12 speed

```
double file_t::speed = 0
```
Definition at line 1201 of file rtklib.h.

12.111.2.13 start

```
double file_t::start = 0
```
Definition at line 1200 of file rtklib.h.

12.111.2.14 swapintv

```
double file_t::swapintv = 0
```
Definition at line 1202 of file rtklib.h.

12.111.2.15 tick

```
unsigned int file_t::tick = 0
```
Definition at line 1197 of file rtklib.h.

12.111.2.16 tick_f

```
unsigned int file_t::tick_f = 0
```
Definition at line 1198 of file rtklib.h.

12.111.2.17 time

```
gtime_t file_t::time = {}
```
Definition at line 1195 of file rtklib.h.

12.111.2.18 timetag

```
int file_t::timetag
```
Definition at line 1192 of file rtklib.h.

12.111.2.19 wtime

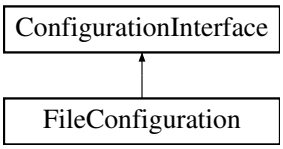
```
gtime_t file_t::wtime = {}
```

Definition at line 1196 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.112 FileConfiguration Class Reference

This class is an implementation of the interface ConfigurationInterface.
`#include <file_configuration.h>`
Inheritance diagram for FileConfiguration:



Public Member Functions

- **FileConfiguration** (std::string filename)
- std::string property (std::string property_name, std::string default_value) const override
- bool property (std::string property_name, bool default_value) const override
- int64_t property (std::string property_name, int64_t default_value) const override
- uint64_t property (std::string property_name, uint64_t default_value) const override
- int32_t property (std::string property_name, int32_t default_value) const override
- uint32_t property (std::string property_name, uint32_t default_value) const override
- int16_t property (std::string property_name, int16_t default_value) const override
- uint16_t property (std::string property_name, uint16_t default_value) const override
- float property (std::string property_name, float default_value) const override
- double property (std::string property_name, double default_value) const override
- void set_property (std::string property_name, std::string value) override
- bool is_present (const std::string &property_name) const override
- bool **has_section** () const

12.112.1 Detailed Description

This class is an implementation of the interface ConfigurationInterface.
Derived from ConfigurationInterface, this class implements an interface to a configuration file. This implementation has a text file as the source for the values of the parameters. The file is in the INI format, containing sections and pairs of names and values. For more information about the INI format, see https://en.wikipedia.org/↩wiki/INI_file
Definition at line 48 of file file_configuration.h.

12.112.2 Member Function Documentation

12.112.2.1 is_present()

```
bool FileConfiguration::is_present (
            const std::string & property_name) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.2 property() [1/10]

```
bool FileConfiguration::property (
            std::string property_name,
            bool default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.3 property() [2/10]

```
double FileConfiguration::property (
            std::string property_name,
            double default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.4 property() [3/10]

```
float FileConfiguration::property (
            std::string property_name,
            float default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.5 property() [4/10]

```
int16_t FileConfiguration::property (
            std::string property_name,
            int16_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.6 property() [5/10]

```
int32_t FileConfiguration::property (
            std::string property_name,
            int32_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.7 property() [6/10]

```
int64_t FileConfiguration::property (
            std::string property_name,
            int64_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.8 property() [7/10]

```
std::string FileConfiguration::property (
            std::string property_name,
            std::string default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.9 property() [8/10]

```
uint16_t FileConfiguration::property (
            std::string property_name,
            uint16_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.10 property() [9/10]

```
uint32_t FileConfiguration::property (
            std::string property_name,
            uint32_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.11 property() [10/10]

```
uint64_t FileConfiguration::property (
            std::string property_name,
            uint64_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.112.2.12 set_property()

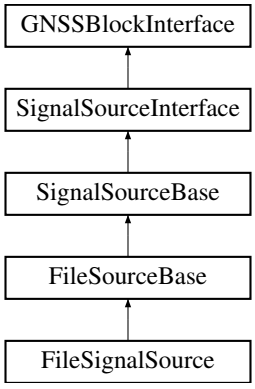
```
void FileConfiguration::set_property (
            std::string property_name,
            std::string value)  [override], [virtual]
```

Implements ConfigurationInterface.
The documentation for this class was generated from the following file:

- file_configuration.h

12.113 FileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <file_signal_source.h>`
Inheritance diagram for FileSignalSource:



Public Member Functions

- **FileSignalSource** (ConfigurationInterface const *configuration, std::string const &role, unsigned int in_↩ streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

The item type.

- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual std::tuple< size_t, bool > itemTypeToSize ()

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- virtual double packetsPerSample () const

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual gnss_shared_ptr< gr::block > source () const

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()

- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_file_source_hook** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **pre_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **pre_disconnect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

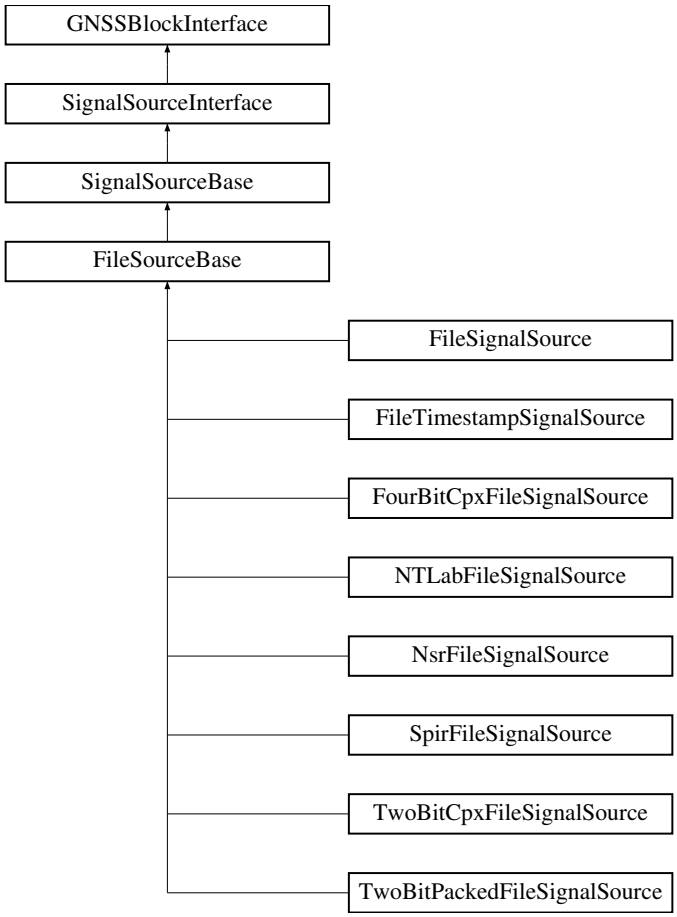
12.113.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 41 of file file_signal_source.h.
The documentation for this class was generated from the following file:

- file_signal_source.h

12.114 FileSourceBase Class Reference

Base class to file-oriented SignalSourceBase GNSS blocks.
`#include <file_source_base.h>`
Inheritance diagram for FileSourceBase:



Public Member Functions

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual std::tuple< size_t, bool > itemTypeToSize ()

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- virtual double packetsPerSample () const

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual gnss_shared_ptr< gr::block > source () const

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_file_source_hook** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **pre_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **pre_disconnect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.114.1 Detailed Description

Base class to file-oriented SignalSourceBase GNSS blocks.
This class supports the following properties:
.filename - the path to the input file

- may be overridden by the -signal_source or -s command-line arguments

.samples - number of samples to process (default 0)

- if not specified or 0, read the entire file; otherwise stop after that many samples

.sampling_frequency - the frequency of the sampled data (samples/second)
.item_type - data type of the samples (default "short")
.header_size - the size of a prefixed header to skip in "samples" (default 0)
.seconds_to_skip - number of seconds of lead-in data to skip over (default 0)
.enable_throttle_control - whether to stop reading if the upstream buffer is full (default false)
.repeat - whether to rewind and continue at end of file (default false)
(probably abstracted to the base class)
.dump - whether to archive input data
.dump_filename - if dumping, path to file for output
Definition at line 68 of file file_source_base.h.

12.114.2 Constructor & Destructor Documentation

12.114.2.1 FileSourceBase()

```
FileSourceBase::FileSourceBase (
            ConfigurationInterface const * configuration,
            std::string const & role,
            std::string impl,
            Concurrent_Queue< pmt::pmt_t > * queue,
            std::string default_item_type = "short")  [protected]
```

Constructor.
Subclasses may want to assert default item types that are appropriate to the specific file type supported. Rather than require the item type to be specified in the config file, allow sub-classes to impose their will

12.114.3 Member Function Documentation

12.114.3.1 computeSamplesInFile()

```
uint64_t FileSourceBase::computeSamplesInFile () const  [protected]
```

Compute the number of samples in the file.

12.114.3.2 connect()

```
void FileSourceBase::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.114.3.3 disconnect()

```
void FileSourceBase::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.114.3.4 filename()

```
std::string FileSourceBase::filename () const
```

The file to read.

12.114.3.5 get_left_block()

```
gr::basic_block_sptr FileSourceBase::get_left_block () [override], [virtual]
```
Implements GNSSBlockInterface.

12.114.3.6 get_right_block()

```
gr::basic_block_sptr FileSourceBase::get_right_block () [override], [virtual]
```
Implements GNSSBlockInterface.

12.114.3.7 init()

```
void FileSourceBase::init () [protected]
```
Perform post-construction initialization.

12.114.3.8 item_size()

```
size_t FileSourceBase::item_size () [override], [virtual]
```
The configured size of each item.
Implements GNSSBlockInterface.
References item_size().
Referenced by item_size().

12.114.3.9 item_type()

```
std::string FileSourceBase::item_type () const
```
The item type.

12.114.3.10 itemTypeToSize()

```
virtual std::tuple< size_t, bool > FileSourceBase::itemTypeToSize () [protected], [virtual]
```
Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented in FourBitCpxFileSignalSource, NsrFileSignalSource, NTLabFileSignalSource, SpirFileSignalSource, TwoBitCpxFileSignalSource, and TwoBitPackedFileSignalSource.

12.114.3.11 packetsPerSample()

```
virtual double FileSourceBase::packetsPerSample () const [protected], [virtual]
```
The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented in FourBitCpxFileSignalSource, NsrFileSignalSource, NTLabFileSignalSource, TwoBitCpxFileSignalSource, and TwoBitPackedFileSignalSource.

12.114.3.12 repeat()

```
bool FileSourceBase::repeat () const
```
Whether to repeat reading after end-of-file.

12.114.3.13 samples()

```
uint64_t FileSourceBase::samples () const
```
The number of samples in the file.

12.114.3.14 samplesToSkip()

```
virtual uint64_t FileSourceBase::samplesToSkip () const [protected], [virtual]
```
Compute the number of samples to skip.

12.114.3.15 sampling_frequency()

```
int64_t FileSourceBase::sampling_frequency () const
```
The sampling frequency of the source file.

12.114.3.16 source()

`virtual gnss_shared_ptr< gr::block > FileSourceBase::source () const [protected], [virtual]`
Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented in FileTimestampSignalSource, FourBitCpxFileSignalSource, NsrFileSignalSource, NTLabFileSignalSource, SpirFileSignalSource, TwoBitCpxFileSignalSource, and TwoBitPackedFileSignalSource.

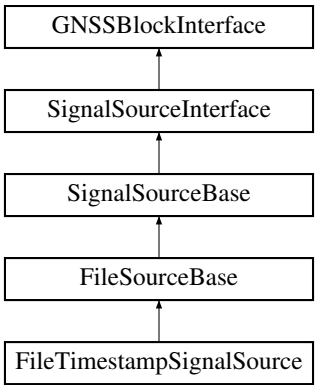
12.114.3.17 source_item_size()

`virtual size_t FileSourceBase::source_item_size () const [protected], [virtual]`
For complex source chains, the size of the file item may not be the same as the size of the.
The documentation for this class was generated from the following file:

- file_source_base.h

12.115 FileTimestampSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <file_timestamp_signal_source.h>`
Inheritance diagram for FileTimestampSignalSource:



Public Member Functions

- **FileTimestampSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

The sampling frequency of the source file.

- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual std::tuple< size_t, bool > itemTypeToSize ()

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- virtual double packetsPerSample () const

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()

- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.115.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 37 of file file_timestamp_signal_source.h.

12.115.2 Member Function Documentation

12.115.2.1 create_file_source_hook()

```
void FileTimestampSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.115.2.2 pre_connect_hook()

```
void FileTimestampSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.115.2.3 pre_disconnect_hook()

```
void FileTimestampSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.115.2.4 source()

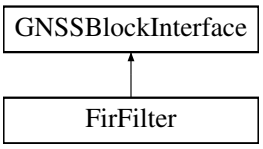
```
gnss_shared_ptr< gr::block > FileTimestampSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- file_timestamp_signal_source.h

12.116 FirFilter Class Reference

This class adapts a GNU Radio gr_fir_filter designed with pm_remez.
`#include <fir_filter.h>`
Inheritance diagram for FirFilter:



Public Member Functions

- FirFilter (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)

  Constructor.
- ~FirFilter ()=default

  Destructor.
- std::string role () override
- std::string implementation () override

  Returns "Fir_Filter".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.116.1 Detailed Description

This class adapts a GNU Radio gr_fir_filter designed with pm_remez.
See Parks-McClellan FIR filter design, https://en.wikipedia.org/wiki/Parks-McClellan_filter_design_algorithm Calculates the optimal (in the Chebyshev/minimax sense) FIR filter impulse response given a set of band edges, the desired response on those bands, and the weight given to the error in those bands.
Definition at line 59 of file fir_filter.h.

12.116.2 Constructor & Destructor Documentation

12.116.2.1 FirFilter()

```
FirFilter::FirFilter (
            const ConfigurationInterface * configuration,
            std::string role,
            unsigned int in_streams,
            unsigned int out_streams)
```

Constructor.

12.116.2.2 ~FirFilter()

```
FirFilter::~FirFilter ()  [default]
```

Destructor.

12.116.3 Member Function Documentation

12.116.3.1 connect()

```
void FirFilter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.116.3.2 disconnect()

```
void FirFilter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.116.3.3 get_left_block()

```
gr::basic_block_sptr FirFilter::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.116.3.4 get_right_block()

```
gr::basic_block_sptr FirFilter::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.116.3.5 implementation()

```
std::string FirFilter::implementation ()  [inline], [override], [virtual]
```

Returns "Fir_Filter".
Implements GNSSBlockInterface.
Definition at line 77 of file fir_filter.h.

12.116.3.6 item_size()

```
size_t FirFilter::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 82 of file fir_filter.h.

12.116.3.7 role()

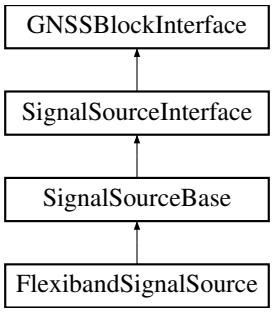
```
std::string FirFilter::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 71 of file fir_filter.h.
The documentation for this class was generated from the following file:

- fir_filter.h

12.117 FlexibandSignalSource Class Reference

This class configures and reads samples from Teleorbit Flexiband front-end. This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR).
`#include <flexiband_signal_source.h>`
Inheritance diagram for FlexibandSignalSource:



Public Member Functions

- **FlexibandSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.117.1 Detailed Description

This class configures and reads samples from Teleorbit Flexiband front-end. This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR).
Definition at line 48 of file flexiband_signal_source.h.

12.117.2 Member Function Documentation

12.117.2.1 connect()

```
void FlexibandSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.117.2.2 disconnect()

```
void FlexibandSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.117.2.3 get_left_block()

```
gr::basic_block_sptr FlexibandSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.117.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr FlexibandSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.117.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr FlexibandSignalSource::get_right_block (
            int RF_channel)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.117.2.6 item_size()

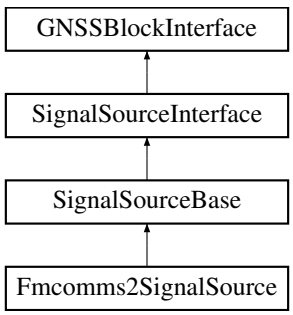
```
size_t FlexibandSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 57 of file flexiband_signal_source.h.
The documentation for this class was generated from the following file:

- flexiband_signal_source.h

12.118 Fmcomms2SignalSource Class Reference

Inheritance diagram for Fmcomms2SignalSource:



Public Member Functions

- **Fmcomms2SignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.118.1 Detailed Description

Definition at line 44 of file fmcomms2_signal_source.h.

12.118.2 Member Function Documentation

12.118.2.1 connect()

```
void Fmcomms2SignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.118.2.2 disconnect()

```
void Fmcomms2SignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.118.2.3 get_left_block()

```
gr::basic_block_sptr Fmcomms2SignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.118.2.4 get_right_block()

```
gr::basic_block_sptr Fmcomms2SignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.118.2.5 item_size()

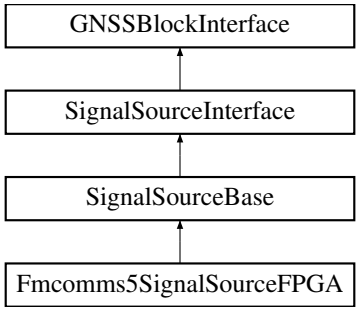
```
size_t Fmcomms2SignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 53 of file fmcomms2_signal_source.h.
The documentation for this class was generated from the following file:

- fmcomms2_signal_source.h

12.119 Fmcomms5SignalSourceFPGA Class Reference

Inheritance diagram for Fmcomms5SignalSourceFPGA:



Public Member Functions

- **Fmcomms5SignalSourceFPGA** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.119.1 Detailed Description

Definition at line 46 of file fmcomms5_signal_source_fpga.h.

12.119.2 Member Function Documentation

12.119.2.1 connect()

```
void Fmcomms5SignalSourceFPGA::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.119.2.2 disconnect()

```
void Fmcomms5SignalSourceFPGA::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.119.2.3 get_left_block()

```
gr::basic_block_sptr Fmcomms5SignalSourceFPGA::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.119.2.4 get_right_block()

```
gr::basic_block_sptr Fmcomms5SignalSourceFPGA::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.119.2.5 item_size()

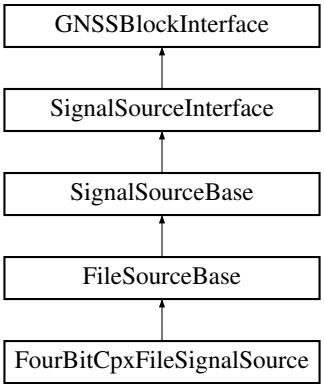
```
size_t Fmcomms5SignalSourceFPGA::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 55 of file fmcomms5_signal_source_fpga.h.
The documentation for this class was generated from the following file:

- fmcomms5_signal_source_fpga.h

12.120 FourBitCpxFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <four_bit_cpx_file_signal_source.h>`
Inheritance diagram for FourBitCpxFileSignalSource:



Public Member Functions

- **FourBitCpxFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- double packetsPerSample () const override

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.

- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.120.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 43 of file four_bit_cpx_file_signal_source.h.

12.120.2 Member Function Documentation

12.120.2.1 create_file_source_hook()

```
void FourBitCpxFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.120.2.2 itemTypeToSize()

```
std::tuple< size_t, bool > FourBitCpxFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```

Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.120.2.3 packetsPerSample()

```
double FourBitCpxFileSignalSource::packetsPerSample () const  [override], [protected], [virtual]
```

The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented from FileSourceBase.

12.120.2.4 pre_connect_hook()

```
void FourBitCpxFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.120.2.5 pre_disconnect_hook()

```
void FourBitCpxFileSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.120.2.6 source()

```
gnss_shared_ptr< gr::block > FourBitCpxFileSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- four_bit_cpx_file_signal_source.h

12.121 Fpga_Acquisition Class Reference

Class that implements carrier wipe-off and correlators.
`#include <fpga_acquisition.h>`

Public Member Functions

- Fpga_Acquisition (std::string device_name, uint32_t select_queue, std::vector< std::pair< uint32_t, uint32_t > > &downsampling_filter_specs, uint32_t &max_FFT_size)

  Constructor.
- ~Fpga_Acquisition ()=default

  Destructor.
- void init (uint32_t nsamples, uint32_t d_fft_size, int64_t resampled_fs, uint32_t downsampling_filter_num, uint32_t excludelimit, uint32_t *all_fft_codes)

  Initialize acquisition parameters.
- bool set_local_code (uint32_t PRN)

  Select the code with the chosen PRN.
- void set_doppler_sweep (uint32_t num_sweeps, uint32_t doppler_step, int32_t doppler_min)

  Configure the doppler sweep parameters in the FPGA.
- void run_acquisition ()

  Run the acquisition process in the FPGA.
- void read_acquisition_results (uint32_t *max_index, float *firstpeak, float *secondpeak, uint64_t *initial_sample, float *power_sum, uint32_t *doppler_index, uint32_t *total_blk_exp)

  Read the results of the acquisition process.
- void reset_acquisition ()

  Reset the FPGA PL.
- void stop_acquisition ()

  stop the acquisition and the FPGA modules.
- void set_block_exp (uint32_t total_block_exp)

  Set the block exponent of the FFT in the FPGA.
- void write_local_code (void)

  Write the PRN code in the FPGA.

- void configure_acquisition (void)

  Write the acquisition parameters into the FPGA.
- void open_device ()

  Open the device driver.
- void close_device ()

  Close the device driver.

12.121.1 Detailed Description

Class that implements carrier wipe-off and correlators.
Definition at line 39 of file fpga_acquisition.h.

12.121.2 Constructor & Destructor Documentation

12.121.2.1 Fpga_Acquisition()

```
Fpga_Acquisition::Fpga_Acquisition (
            std::string device_name,
            uint32_t select_queue,
            std::vector< std::pair< uint32_t, uint32_t > > & downsampling_filter_specs,
            uint32_t & max_FFT_size)
```

Constructor.

12.121.2.2 ~Fpga_Acquisition()

```
Fpga_Acquisition::~Fpga_Acquisition ()  [default]
```

Destructor.

12.121.3 Member Function Documentation

12.121.3.1 close_device()

```
void Fpga_Acquisition::close_device ()
```

Close the device driver.

12.121.3.2 configure_acquisition()

```
void Fpga_Acquisition::configure_acquisition (
            void )
```

Write the acquisition parameters into the FPGA.

12.121.3.3 init()

```
void Fpga_Acquisition::init (
            uint32_t nsamples,
            uint32_t d_fft_size,
            int64_t resampled_fs,
            uint32_t downsampling_filter_num,
            uint32_t excludelimit,
            uint32_t * all_fft_codes)
```

Initialize acquisition parameters.

12.121.3.4 open_device()

```
void Fpga_Acquisition::open_device ()
```

Open the device driver.

12.121.3.5 read_acquisition_results()

```
void Fpga_Acquisition::read_acquisition_results (
            uint32_t * max_index,
            float * firstpeak,
            float * secondpeak,
            uint64_t * initial_sample,
            float * power_sum,
            uint32_t * doppler_index,
            uint32_t * total_blk_exp)
```

Read the results of the acquisition process.

12.121.3.6 reset_acquisition()

```
void Fpga_Acquisition::reset_acquisition ()
```

Reset the FPGA PL.

12.121.3.7 run_acquisition()

```
void Fpga_Acquisition::run_acquisition ()
```

Run the acquisition process in the FPGA.

12.121.3.8 set_block_exp()

```
void Fpga_Acquisition::set_block_exp (
            uint32_t total_block_exp)
```

Set the block exponent of the FFT in the FPGA.

12.121.3.9 set_doppler_sweep()

```
void Fpga_Acquisition::set_doppler_sweep (
            uint32_t num_sweeps,
            uint32_t doppler_step,
            int32_t doppler_min)
```

Configure the doppler sweep parameters in the FPGA.

12.121.3.10 set_local_code()

```
bool Fpga_Acquisition::set_local_code (
            uint32_t PRN)
```

Select the code with the chosen PRN.

12.121.3.11 stop_acquisition()

```
void Fpga_Acquisition::stop_acquisition ()
```

stop the acquisition and the FPGA modules.

12.121.3.12 write_local_code()

```
void Fpga_Acquisition::write_local_code (
            void )
```

Write the PRN code in the FPGA.

The documentation for this class was generated from the following file:

- fpga_acquisition.h

12.122 Fpga_buffer_monitor Class Reference

Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers.

`#include <fpga_buffer_monitor.h>`

Public Member Functions

- Fpga_buffer_monitor (uint32_t num_freq_bands, bool dump, std::string dump_filename)

  Constructor.
- ~Fpga_buffer_monitor ()

  Destructor.
- bool check_buffer_overflow_and_monitor_buffer_status ()

  This function checks buffer overflow and monitors the FPGA buffer status.

12.122.1 Detailed Description

Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers.
Definition at line 42 of file fpga_buffer_monitor.h.

12.122.2 Constructor & Destructor Documentation

12.122.2.1 Fpga_buffer_monitor()

```
Fpga_buffer_monitor::Fpga_buffer_monitor (
            uint32_t num_freq_bands,
            bool dump,
            std::string dump_filename)  [explicit]
```

Constructor.

12.122.2.2 ~Fpga_buffer_monitor()

```
Fpga_buffer_monitor::~Fpga_buffer_monitor ()
```

Destructor.

12.122.3 Member Function Documentation

12.122.3.1 check_buffer_overflow_and_monitor_buffer_status()

```
bool Fpga_buffer_monitor::check_buffer_overflow_and_monitor_buffer_status ()
```

This function checks buffer overflow and monitors the FPGA buffer status.
The documentation for this class was generated from the following file:

- fpga_buffer_monitor.h

12.123 Fpga_DMA Class Reference

Class that controls DMA transfers to the FPGA.
`#include <fpga_dma-proxy.h>`

Public Member Functions

- Fpga_DMA ()=default

  Default constructor.
- ~Fpga_DMA ()

  Release any open DMA resources.
- **Fpga_DMA** (const Fpga_DMA &)=delete
- Fpga_DMA & **operator=** (const Fpga_DMA &)=delete
- int DMA_open (void)

  Open the DMA device driver.
- int8_t * get_buffer_address () const

  Obtain DMA buffer address.
- uint32_t get_buffer_size () const

Obtain DMA buffer size.

- int DMA_write (int nbytes) const

  Transfer DMA data.

- int DMA_close (void)

  Close the DMA device driver.

12.123.1 Detailed Description

Class that controls DMA transfers to the FPGA.
Definition at line 26 of file fpga_dma-proxy.h.

12.123.2 Constructor & Destructor Documentation

12.123.2.1 Fpga_DMA()

```
Fpga_DMA::Fpga_DMA ()  [default]
```

Default constructor.
Referenced by ~Fpga_DMA().

12.123.2.2 ~Fpga_DMA()

```
Fpga_DMA::~Fpga_DMA ()
```

Release any open DMA resources.
References Fpga_DMA().

12.123.3 Member Function Documentation

12.123.3.1 DMA_close()

```
int Fpga_DMA::DMA_close (
            void )
```

Close the DMA device driver.

12.123.3.2 DMA_open()

```
int Fpga_DMA::DMA_open (
            void )
```

Open the DMA device driver.

12.123.3.3 DMA_write()

```
int Fpga_DMA::DMA_write (
            int nbytes) const
```

Transfer DMA data.

12.123.3.4 get_buffer_address()

```
int8_t * Fpga_DMA::get_buffer_address () const
```

Obtain DMA buffer address.

12.123.3.5 get_buffer_size()

```
uint32_t Fpga_DMA::get_buffer_size () const
```

Obtain DMA buffer size.
The documentation for this class was generated from the following file:

- fpga_dma-proxy.h

12.124 Fpga_dynamic_bit_selection Class Reference

Controls dynamic bit selection in the FPGA.
`#include <fpga_dynamic_bit_selection.h>`

Public Member Functions

- Fpga_dynamic_bit_selection (bool enable_rx1_band, bool enable_rx2_band)

  Constructor.
- ~Fpga_dynamic_bit_selection ()

  Destructor.
- **Fpga_dynamic_bit_selection** (const Fpga_dynamic_bit_selection &)=delete
- Fpga_dynamic_bit_selection & **operator=** (const Fpga_dynamic_bit_selection &)=delete
- void bit_selection (void)

  Adjusts the bit shift for each enabled frequency band based on signal power.

12.124.1 Detailed Description

Controls dynamic bit selection in the FPGA.
Definition at line 38 of file fpga_dynamic_bit_selection.h.

12.124.2 Constructor & Destructor Documentation

12.124.2.1 Fpga_dynamic_bit_selection()

```
Fpga_dynamic_bit_selection::Fpga_dynamic_bit_selection (
            bool enable_rx1_band,
            bool enable_rx2_band)  [explicit]
```

Constructor.
Referenced by ~Fpga_dynamic_bit_selection().

12.124.2.2 ~Fpga_dynamic_bit_selection()

```
Fpga_dynamic_bit_selection::~Fpga_dynamic_bit_selection ()
```

Destructor.
References Fpga_dynamic_bit_selection().

12.124.3 Member Function Documentation

12.124.3.1 bit_selection()

```
void Fpga_dynamic_bit_selection::bit_selection (
            void )
```

Adjusts the bit shift for each enabled frequency band based on signal power.
The documentation for this class was generated from the following file:

- fpga_dynamic_bit_selection.h

12.125 Fpga_Multicorrelator_8sc Class Reference

Class that implements carrier wipe-off and correlators.
`#include <fpga_multicorrelator.h>`

Public Member Functions

- Fpga_Multicorrelator_8sc (int32_t n_correlators, int32_t *ca_codes, int32_t *data_codes, uint32_t code_↩ length_chips, bool track_pilot, uint32_t code_samples_per_chip)

  Constructor.

- ~Fpga_Multicorrelator_8sc ()

  Destructor.
- void set_output_vectors (gr_complex *corr_out, gr_complex *Prompt_Data)

  Configure pointers to the FPGA multicorrelator results.
- void set_local_code_and_taps (float *shifts_chips, float *prompt_data_shift, int32_t PRN)

  Configure the local code in the FPGA multicorrelator.
- void update_local_code ()

  Configure code phase and code rate parameters in the FPGA.
- void Carrier_wipeoff_multicorrelator_resampler (float rem_carrier_phase_in_rad, float phase_step_rad, float carrier_phase_rate_step_rad, float rem_code_phase_chips, float code_phase_step_chips, float code_↩ phase_rate_step_chips, int32_t signal_length_samples)

  Perform a multicorrelation.
- bool free ()

  Stop the correlation process in the FPGA and free code phase and code rate parameters.
- void open_channel (const std::string &device_io_name, uint32_t channel)

  Open the FPGA device driver.
- void set_initial_sample (uint64_t samples_offset)

  Set the initial sample number where the tracking process begins.
- uint64_t read_sample_counter ()

  Read the sample counter in the FPGA.
- void lock_channel ()

  Start the tracking process in the FPGA.
- void unlock_channel ()

  finish the tracking process in the FPGA
- void set_secondary_code_lengths (uint32_t secondary_code_0_length, uint32_t secondary_code_1_length)

  Set the secondary code length in the FPGA. This is only used when extended coherent integration is enabled in the FPGA. If tracking the pilot is enabled then secondary_code_0_length is the length of the pilot secondary code and secondary_code_1_length is the length of the data secondary code. If tracking the pilot is disabled then secondary↩ _code_0_length is the length of the data secondary code, and secondary_code_1_length must be set to zero.
- void initialize_secondary_code (uint32_t secondary_code, std::string *secondary_code_string)

  Initialize the secondary code in the FPGA. If tracking the pilot is enabled then the pilot secondary code is configured when secondary_code = 0 and the data secondary code is configured when secondary_code = 1. If tracking the pilot is disabled then the data secondary code is configured when secondary code = 0.
- void update_prn_code_length (uint32_t first_prn_length, uint32_t next_prn_length)

  Set the PRN length in the FPGA in number of samples. This function is only used then extended coherent integration is enabled in the FPGA. The FPGA allows for the configuration of two PRN lengths. When the length of the extended coherent integration is bigger than the length of the PRN code, the FPGA uses the first_length_secondary_code as the length of the PRN code immediately following the beginning of the extended coherent integration, and the next↩ _length_secondary_code as the length of the remaining PRN codes. The purpose of this is to have the option to allow the FPGA to compensate for a possible deviation between the nominal value of the PRN code length and the measured PRN code length in the PRN immediately following the start of the coherent integration only. If this option is not used then write the same value to first_length_secondary_code and next_length_secondary_code.
- void enable_secondary_codes ()

  Enable the use of secondary codes in the FPGA.
- void disable_secondary_codes ()

  Disable the use of secondary codes in the FPGA.

12.125.1 Detailed Description

Class that implements carrier wipe-off and correlators.
Definition at line 40 of file fpga_multicorrelator.h.

12.125.2 Constructor & Destructor Documentation

12.125.2.1 Fpga_Multicorrelator_8sc()

```
Fpga_Multicorrelator_8sc::Fpga_Multicorrelator_8sc (
            int32_t n_correlators,
            int32_t * ca_codes,
            int32_t * data_codes,
            uint32_t code_length_chips,
            bool track_pilot,
            uint32_t code_samples_per_chip)
```

Constructor.

12.125.2.2 ~Fpga_Multicorrelator_8sc()

```
Fpga_Multicorrelator_8sc::~Fpga_Multicorrelator_8sc ()
```

Destructor.

12.125.3 Member Function Documentation

12.125.3.1 Carrier_wipeoff_multicorrelator_resampler()

```
void Fpga_Multicorrelator_8sc::Carrier_wipeoff_multicorrelator_resampler (
            float rem_carrier_phase_in_rad,
            float phase_step_rad,
            float carrier_phase_rate_step_rad,
            float rem_code_phase_chips,
            float code_phase_step_chips,
            float code_phase_rate_step_chips,
            int32_t signal_length_samples)
```

Perform a multicorrelation.

12.125.3.2 disable_secondary_codes()

```
void Fpga_Multicorrelator_8sc::disable_secondary_codes ()
```

Disable the use of secondary codes in the FPGA.

12.125.3.3 enable_secondary_codes()

```
void Fpga_Multicorrelator_8sc::enable_secondary_codes ()
```

Enable the use of secondary codes in the FPGA.

12.125.3.4 free()

```
bool Fpga_Multicorrelator_8sc::free ()
```

Stop the correlation process in the FPGA and free code phase and code rate parameters.

12.125.3.5 initialize_secondary_code()

```
void Fpga_Multicorrelator_8sc::initialize_secondary_code (
            uint32_t secondary_code,
            std::string * secondary_code_string)
```

Initialize the secondary code in the FPGA. If tracking the pilot is enabled then the pilot secondary code is configured when secondary_code = 0 and the data secondary code is configured when secondary_code = 1. If tracking the pilot is disabled then the data secondary code is configured when secondary code = 0.

12.125.3.6 lock_channel()

```
void Fpga_Multicorrelator_8sc::lock_channel ()
```

Start the tracking process in the FPGA.

12.125.3.7 open_channel()

```
void Fpga_Multicorrelator_8sc::open_channel (
            const std::string & device_io_name,
            uint32_t channel)
```

Open the FPGA device driver.

12.125.3.8 read_sample_counter()

```
uint64_t Fpga_Multicorrelator_8sc::read_sample_counter ()
```

Read the sample counter in the FPGA.

12.125.3.9 set_initial_sample()

```
void Fpga_Multicorrelator_8sc::set_initial_sample (
            uint64_t samples_offset)
```

Set the initial sample number where the tracking process begins.

12.125.3.10 set_local_code_and_taps()

```
void Fpga_Multicorrelator_8sc::set_local_code_and_taps (
            float * shifts_chips,
            float * prompt_data_shift,
            int32_t PRN)
```

Configure the local code in the FPGA multicorrelator.

12.125.3.11 set_output_vectors()

```
void Fpga_Multicorrelator_8sc::set_output_vectors (
            gr_complex * corr_out,
            gr_complex * Prompt_Data)
```

Configure pointers to the FPGA multicorrelator results.

12.125.3.12 set_secondary_code_lengths()

```
void Fpga_Multicorrelator_8sc::set_secondary_code_lengths (
            uint32_t secondary_code_0_length,
            uint32_t secondary_code_1_length)
```

Set the secondary code length in the FPGA. This is only used when extended coherent integration is enabled in the FPGA. If tracking the pilot is enabled then secondary_code_0_length is the length of the pilot secondary code and secondary_code_1_length is the length of the data secondary code. If tracking the pilot is disabled then secondary_code_0_length is the length of the data secondary code, and secondary_code_1_length must be set to zero.

12.125.3.13 unlock_channel()

```
void Fpga_Multicorrelator_8sc::unlock_channel ()
```

finish the tracking process in the FPGA

12.125.3.14 update_local_code()

```
void Fpga_Multicorrelator_8sc::update_local_code ()
```

Configure code phase and code rate parameters in the FPGA.

12.125.3.15 update_prn_code_length()

```
void Fpga_Multicorrelator_8sc::update_prn_code_length (
            uint32_t first_prn_length,
            uint32_t next_prn_length)
```

Set the PRN length in the FPGA in number of samples. This function is only used then extended coherent integration is enabled in the FPGA. The FPGA allows for the configuration of two PRN lengths. When the length of the extended coherent integration is bigger than the length of the PRN code, the FPGA uses the first_length_secondary_code as the length of the PRN code immediately following the beginning of the extended coherent integration, and the next_length_secondary_code as the length of the remaining PRN codes. The purpose of this is to have the option to allow the FPGA to compensate for a possible deviation between the nominal value of the PRN code length and the measured PRN code length in the PRN immediately following the start of the coherent integration only. If this option is not used then write the same value to first_length_secondary_code and next_length_secondary_code.
The documentation for this class was generated from the following file:

- fpga_multicorrelator.h

12.126 Fpga_spidev Class Reference

Public Member Functions

- Fpga_spidev ()=default
  Default constructor.
- ~Fpga_spidev ()=default
  Default destructor.
- int write_reg32 (char addr, uint32_t data)
  write a register through the SPI.
- int read_reg32 (uint8_t addr, uint32_t *copy_to)
  read a register through the SPI.
- int SPI_open (std::string spi_device_name)
  Open the SPI device driver.
- int SPI_close (void) const
  Close the SPI device driver.

12.126.1 Detailed Description

Definition at line 22 of file fpga_spidev.h.

12.126.2 Constructor & Destructor Documentation

12.126.2.1 Fpga_spidev()

```
Fpga_spidev::Fpga_spidev ()  [default]
```

Default constructor.

12.126.2.2 ~Fpga_spidev()

```
Fpga_spidev::~Fpga_spidev ()  [default]
```

Default destructor.

12.126.3 Member Function Documentation

12.126.3.1 read_reg32()

```
int Fpga_spidev::read_reg32 (
            uint8_t addr,
            uint32_t * copy_to)
```

read a register through the SPI.

12.126.3.2 SPI_close()

```
int Fpga_spidev::SPI_close (
            void ) const
```

Close the SPI device driver.

12.126.3.3 SPI_open()

```
int Fpga_spidev::SPI_open (
            std::string spi_device_name)
```

Open the SPI device driver.

12.126.3.4 write_reg32()

```
int Fpga_spidev::write_reg32 (
            char addr,
            uint32_t data)
```

write a register through the SPI.
The documentation for this class was generated from the following file:

- fpga_spidev.h

12.127 Fpga_Switch Class Reference

Class that controls the switch in the FPGA, which connects the FPGA acquisition and multicorrelator modules to either the DMA or the Analog Front-End.
`#include <fpga_switch.h>`

Public Member Functions

- Fpga_Switch (void)
  Constructor.
- ~Fpga_Switch ()
  Destructor.
- void set_switch_position (int32_t switch_position)
  This function configures the switch in th eFPGA.

12.127.1 Detailed Description

Class that controls the switch in the FPGA, which connects the FPGA acquisition and multicorrelator modules to either the DMA or the Analog Front-End.
Definition at line 39 of file fpga_switch.h.

12.127.2 Constructor & Destructor Documentation

12.127.2.1 Fpga_Switch()

```
Fpga_Switch::Fpga_Switch (
            void )
```

Constructor.

12.127.2.2 ~Fpga_Switch()

```
Fpga_Switch::~Fpga_Switch ()
```

Destructor.

12.127.3 Member Function Documentation

12.127.3.1 set_switch_position()

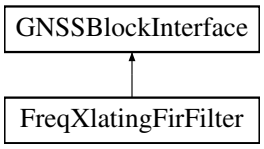
```
void Fpga_Switch::set_switch_position (
            int32_t switch_position)
```

This function configures the switch in th eFPGA.
The documentation for this class was generated from the following file:

- fpga_switch.h

12.128 FreqXlatingFirFilter Class Reference

This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez.
`#include <freq_xlating_fir_filter.h>`
Inheritance diagram for FreqXlatingFirFilter:



Public Member Functions

- **FreqXlatingFirFilter** (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Freq_Xlating_Fir_Filter".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.128.1 Detailed Description

This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez.
Construct a FIR filter with the given taps and a composite frequency translation that shifts intermediate_freq_ down to zero Hz. The frequency translation logically comes before the filtering operation.
See Parks-McClellan FIR filter design, https://en.wikipedia.org/wiki/Parks-McClellan_↩filter_design_algorithm Calculates the optimal (in the Chebyshev/minimax sense) FIR filter impulse response given a set of band edges, the desired response on those bands, and the weight given to the error in those bands.
Definition at line 58 of file freq_xlating_fir_filter.h.

12.128.2 Member Function Documentation

12.128.2.1 connect()

```
void FreqXlatingFirFilter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.128.2.2 disconnect()

```
void FreqXlatingFirFilter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.128.2.3 get_left_block()

```
gr::basic_block_sptr FreqXlatingFirFilter::get_left_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.128.2.4 get_right_block()

```
gr::basic_block_sptr FreqXlatingFirFilter::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.128.2.5 implementation()

```
std::string FreqXlatingFirFilter::implementation ()  [inline], [override], [virtual]
```
Returns "Freq_Xlating_Fir_Filter".
Implements GNSSBlockInterface.
Definition at line 73 of file freq_xlating_fir_filter.h.

12.128.2.6 item_size()

```
size_t FreqXlatingFirFilter::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 78 of file freq_xlating_fir_filter.h.

12.128.2.7 role()

```
std::string FreqXlatingFirFilter::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 67 of file freq_xlating_fir_filter.h.
The documentation for this class was generated from the following file:

- freq_xlating_fir_filter.h

12.129 ftp_t Struct Reference

Public Attributes

- int state
- int proto
- int error
- char addr [1024]
- char file [1024]
- char user [256]
- char passwd [256]
- char local [1024]
- int topts [4]
- gtime_t tnext
- pthread_t thread

12.129.1 Detailed Description

Definition at line 1264 of file rtklib.h.

12.129.2 Member Data Documentation

12.129.2.1 addr

```
char ftp_t::addr[1024]
```
Definition at line 1270 of file rtklib.h.

12.129.2.2 error

`int ftp_t::error`
Definition at line 1268 of file rtklib.h.

12.129.2.3 file

`char ftp_t::file[1024]`
Definition at line 1271 of file rtklib.h.

12.129.2.4 local

`char ftp_t::local[1024]`
Definition at line 1274 of file rtklib.h.

12.129.2.5 passwd

`char ftp_t::passwd[256]`
Definition at line 1273 of file rtklib.h.

12.129.2.6 proto

`int ftp_t::proto`
Definition at line 1267 of file rtklib.h.

12.129.2.7 state

`int ftp_t::state`
Definition at line 1266 of file rtklib.h.

12.129.2.8 thread

`pthread_t ftp_t::thread`
Definition at line 1277 of file rtklib.h.

12.129.2.9 tnext

`gtime_t ftp_t::tnext`
Definition at line 1276 of file rtklib.h.

12.129.2.10 topts

`int ftp_t::topts[4]`
Definition at line 1275 of file rtklib.h.

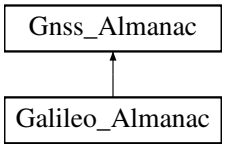
12.129.2.11 user

`char ftp_t::user[256]`
Definition at line 1272 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.130 Galileo_Almanac Class Reference

This class is a storage for the Galileo SV ALMANAC data.
`#include <galileo_almanac.h>`
Inheritance diagram for Galileo_Almanac:



Public Member Functions

- Galileo_Almanac ()
- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

Public Member Functions inherited from Gnss_Almanac

- Gnss_Almanac ()=default
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosVelComputation (double transmitTime, std::array< double, 7 > &pos_vel_dtr) const

  Computes satellite Position and Velocity, in ECEF, for a given time (expressed in seconds of week).

Public Attributes

- int32_t IODa {}
- int32_t E5b_HS {}
- int32_t E1B_HS {}
- int32_t E5a_HS {}

Public Attributes inherited from Gnss_Almanac

- uint32_t PRN {}

  SV PRN NUMBER.
- double delta_i {}

  Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles).
- int32_t toa {}

  Almanac data reference time of week [s].
- int32_t WNa {}

  Almanac week number.
- double M_0 {}

  Mean Anomaly at Reference Time [semi-circles].
- double ecc {}

  Eccentricity [dimensionless].
- double sqrtA {}

  Square Root of the Semi-Major Axis [sqrt(m)].
- double OMEGA_0 {}

  Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].
- double omega {}

  Argument of Perigee [semi-cicles].
- double OMEGAdot {}

  Rate of Right Ascension [semi-circles/s].
- double af0 {}

  Coefficient 0 of code phase offset model [s].
- double af1 {}

Coefficient 1 of code phase offset model [s/s].

- bool from_startup_load {false}

  Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the broadcast. Runtime-only provenance flag, intentionally not serialized.

Additional Inherited Members

Protected Attributes inherited from Gnss_Almanac

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J': QZSS.

12.130.1 Detailed Description

This class is a storage for the Galileo SV ALMANAC data.
Definition at line 33 of file galileo_almanac.h.

12.130.2 Constructor & Destructor Documentation

12.130.2.1 Galileo_Almanac()

```
Galileo_Almanac::Galileo_Almanac ()  [inline]
```

Default constructor
Definition at line 39 of file galileo_almanac.h.
References Gnss_Almanac::System.

12.130.3 Member Function Documentation

12.130.3.1 serialize()

```
template<class Archive>
void Galileo_Almanac::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

Definition at line 51 of file galileo_almanac.h.

12.130.4 Member Data Documentation

12.130.4.1 E1B_HS

```
int32_t Galileo_Almanac::E1B_HS {}
```

Definition at line 46 of file galileo_almanac.h.

12.130.4.2 E5a_HS

```
int32_t Galileo_Almanac::E5a_HS {}
```

Definition at line 47 of file galileo_almanac.h.

12.130.4.3 E5b_HS

```
int32_t Galileo_Almanac::E5b_HS {}
```

Definition at line 45 of file galileo_almanac.h.

12.130.4.4 IODa

```
int32_t Galileo_Almanac::IODa {}
```

Definition at line 44 of file galileo_almanac.h.
The documentation for this class was generated from the following file:

- galileo_almanac.h

12.131 Galileo_Almanac_Helper Class Reference

This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD.
`#include <galileo_almanac_helper.h>`

Public Member Functions

- Galileo_Almanac_Helper ()=default

  Default constructor.
- Galileo_Almanac **get_almanac** (int i) const

Public Attributes

- int32_t IOD_a_7 {}
- int32_t WN_a_7 {}
- int32_t t0a_7 {}
- int32_t SVID1_7 {}
- double DELTA_A_7 {}
- double e_7 {}
- double omega_7 {}
- double delta_i_7 {}
- double Omega0_7 {}
- double Omega_dot_7 {}
- double M0_7 {}
- int32_t IOD_a_8 {}
- double af0_8 {}
- double af1_8 {}
- int32_t E5b_HS_8 {}
- int32_t E1B_HS_8 {}
- int32_t E5a_HS_8 {}
- int32_t SVID2_8 {}
- double DELTA_A_8 {}
- double e_8 {}
- double omega_8 {}
- double delta_i_8 {}
- double Omega0_8 {}
- double Omega_dot_8 {}
- int32_t IOD_a_9 {}
- int32_t WN_a_9 {}
- int32_t t0a_9 {}
- double M0_9 {}
- double af0_9 {}
- double af1_9 {}
- int32_t E5b_HS_9 {}
- int32_t E1B_HS_9 {}
- int32_t E5a_HS_9 {}
- int32_t SVID3_9 {}
- double DELTA_A_9 {}
- double e_9 {}
- double omega_9 {}
- double delta_i_9 {}
- int32_t IOD_a_10 {}
- double Omega0_10 {}
- double Omega_dot_10 {}
- double M0_10 {}

- double af0_10 {}
- double af1_10 {}
- int32_t E5b_HS_10 {}
- int32_t E1B_HS_10 {}
- int32_t E5a_HS_10 {}

12.131.1 Detailed Description

This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD.
See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf paragraph 5.1.10
Definition at line 34 of file galileo_almanac_helper.h.

12.131.2 Constructor & Destructor Documentation

12.131.2.1 Galileo_Almanac_Helper()

```
Galileo_Almanac_Helper::Galileo_Almanac_Helper () [default]
```

Default constructor.

12.131.3 Member Data Documentation

12.131.3.1 af0_10

```
double Galileo_Almanac_Helper::af0_10 {}
```

Definition at line 90 of file galileo_almanac_helper.h.

12.131.3.2 af0_8

```
double Galileo_Almanac_Helper::af0_8 {}
```

Definition at line 56 of file galileo_almanac_helper.h.

12.131.3.3 af0_9

```
double Galileo_Almanac_Helper::af0_9 {}
```

Definition at line 74 of file galileo_almanac_helper.h.

12.131.3.4 af1_10

```
double Galileo_Almanac_Helper::af1_10 {}
```

Definition at line 91 of file galileo_almanac_helper.h.

12.131.3.5 af1_8

```
double Galileo_Almanac_Helper::af1_8 {}
```

Definition at line 57 of file galileo_almanac_helper.h.

12.131.3.6 af1_9

```
double Galileo_Almanac_Helper::af1_9 {}
```

Definition at line 75 of file galileo_almanac_helper.h.

12.131.3.7 DELTA_A_7

```
double Galileo_Almanac_Helper::DELTA_A_7 {}
```

Definition at line 46 of file galileo_almanac_helper.h.

12.131.3.8 DELTA_A_8

```
double Galileo_Almanac_Helper::DELTA_A_8 {}
```
Definition at line 62 of file galileo_almanac_helper.h.

12.131.3.9 DELTA_A_9

```
double Galileo_Almanac_Helper::DELTA_A_9 {}
```
Definition at line 80 of file galileo_almanac_helper.h.

12.131.3.10 delta_i_7

```
double Galileo_Almanac_Helper::delta_i_7 {}
```
Definition at line 49 of file galileo_almanac_helper.h.

12.131.3.11 delta_i_8

```
double Galileo_Almanac_Helper::delta_i_8 {}
```
Definition at line 65 of file galileo_almanac_helper.h.

12.131.3.12 delta_i_9

```
double Galileo_Almanac_Helper::delta_i_9 {}
```
Definition at line 83 of file galileo_almanac_helper.h.

12.131.3.13 E1B_HS_10

```
int32_t Galileo_Almanac_Helper::E1B_HS_10 {}
```
Definition at line 93 of file galileo_almanac_helper.h.

12.131.3.14 E1B_HS_8

```
int32_t Galileo_Almanac_Helper::E1B_HS_8 {}
```
Definition at line 59 of file galileo_almanac_helper.h.

12.131.3.15 E1B_HS_9

```
int32_t Galileo_Almanac_Helper::E1B_HS_9 {}
```
Definition at line 77 of file galileo_almanac_helper.h.

12.131.3.16 E5a_HS_10

```
int32_t Galileo_Almanac_Helper::E5a_HS_10 {}
```
Definition at line 94 of file galileo_almanac_helper.h.

12.131.3.17 E5a_HS_8

```
int32_t Galileo_Almanac_Helper::E5a_HS_8 {}
```
Definition at line 60 of file galileo_almanac_helper.h.

12.131.3.18 E5a_HS_9

```
int32_t Galileo_Almanac_Helper::E5a_HS_9 {}
```
Definition at line 78 of file galileo_almanac_helper.h.

12.131.3.19 E5b_HS_10

```
int32_t Galileo_Almanac_Helper::E5b_HS_10 {}
```
Definition at line 92 of file galileo_almanac_helper.h.

12.131.3.20 E5b_HS_8

```
int32_t Galileo_Almanac_Helper::E5b_HS_8 {}
```
Definition at line 58 of file galileo_almanac_helper.h.

12.131.3.21 E5b_HS_9

```
int32_t Galileo_Almanac_Helper::E5b_HS_9 {}
```
Definition at line 76 of file galileo_almanac_helper.h.

12.131.3.22 e_7

```
double Galileo_Almanac_Helper::e_7 {}
```
Definition at line 47 of file galileo_almanac_helper.h.

12.131.3.23 e_8

```
double Galileo_Almanac_Helper::e_8 {}
```
Definition at line 63 of file galileo_almanac_helper.h.

12.131.3.24 e_9

```
double Galileo_Almanac_Helper::e_9 {}
```
Definition at line 81 of file galileo_almanac_helper.h.

12.131.3.25 IOD_a_10

```
int32_t Galileo_Almanac_Helper::IOD_a_10 {}
```
Definition at line 86 of file galileo_almanac_helper.h.

12.131.3.26 IOD_a_7

```
int32_t Galileo_Almanac_Helper::IOD_a_7 {}
```
Definition at line 42 of file galileo_almanac_helper.h.

12.131.3.27 IOD_a_8

```
int32_t Galileo_Almanac_Helper::IOD_a_8 {}
```
Definition at line 55 of file galileo_almanac_helper.h.

12.131.3.28 IOD_a_9

```
int32_t Galileo_Almanac_Helper::IOD_a_9 {}
```
Definition at line 70 of file galileo_almanac_helper.h.

12.131.3.29 M0_10

```
double Galileo_Almanac_Helper::M0_10 {}
```
Definition at line 89 of file galileo_almanac_helper.h.

12.131.3.30 M0_7

```
double Galileo_Almanac_Helper::M0_7 {}
```
Definition at line 52 of file galileo_almanac_helper.h.

12.131.3.31 M0_9

```
double Galileo_Almanac_Helper::M0_9 {}
```
Definition at line 73 of file galileo_almanac_helper.h.

12.131.3.32 Omega0_10

```
double Galileo_Almanac_Helper::Omega0_10 {}
```
Definition at line 87 of file galileo_almanac_helper.h.

12.131.3.33 Omega0_7

```
double Galileo_Almanac_Helper::Omega0_7 {}
```
Definition at line 50 of file galileo_almanac_helper.h.

12.131.3.34 Omega0_8

```
double Galileo_Almanac_Helper::Omega0_8 {}
```
Definition at line 66 of file galileo_almanac_helper.h.

12.131.3.35 omega_7

```
double Galileo_Almanac_Helper::omega_7 {}
```
Definition at line 48 of file galileo_almanac_helper.h.

12.131.3.36 omega_8

```
double Galileo_Almanac_Helper::omega_8 {}
```
Definition at line 64 of file galileo_almanac_helper.h.

12.131.3.37 omega_9

```
double Galileo_Almanac_Helper::omega_9 {}
```
Definition at line 82 of file galileo_almanac_helper.h.

12.131.3.38 Omega_dot_10

```
double Galileo_Almanac_Helper::Omega_dot_10 {}
```
Definition at line 88 of file galileo_almanac_helper.h.

12.131.3.39 Omega_dot_7

```
double Galileo_Almanac_Helper::Omega_dot_7 {}
```
Definition at line 51 of file galileo_almanac_helper.h.

12.131.3.40 Omega_dot_8

```
double Galileo_Almanac_Helper::Omega_dot_8 {}
```
Definition at line 67 of file galileo_almanac_helper.h.

12.131.3.41 SVID1_7

```
int32_t Galileo_Almanac_Helper::SVID1_7 {}
```
Definition at line 45 of file galileo_almanac_helper.h.

12.131.3.42 SVID2_8

```
int32_t Galileo_Almanac_Helper::SVID2_8 {}
```
Definition at line 61 of file galileo_almanac_helper.h.

12.131.3.43 SVID3_9

```
int32_t Galileo_Almanac_Helper::SVID3_9 {}
```
Definition at line 79 of file galileo_almanac_helper.h.

12.131.3.44 t0a_7

```
int32_t Galileo_Almanac_Helper::t0a_7 {}
```
Definition at line 44 of file galileo_almanac_helper.h.

12.131.3.45 t0a_9

```
int32_t Galileo_Almanac_Helper::t0a_9 {}
```
Definition at line 72 of file galileo_almanac_helper.h.

12.131.3.46 WN_a_7

```
int32_t Galileo_Almanac_Helper::WN_a_7 {}
```
Definition at line 43 of file galileo_almanac_helper.h.

12.131.3.47 WN_a_9

```
int32_t Galileo_Almanac_Helper::WN_a_9 {}
```
Definition at line 71 of file galileo_almanac_helper.h.
The documentation for this class was generated from the following file:

- galileo_almanac_helper.h

12.132 Galileo_Cnav_Message Class Reference

This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
`#include <galileo_cnav_message.h>`

Public Member Functions

- void **read_HAS_page** (const std::string &page_string)
- bool is_HAS_in_test_mode () const
- bool is_HAS_page_dummy () const
- bool have_new_HAS_page () const
- Galileo_HAS_page get_HAS_encoded_page () const
- bool get_flag_CRC_test () const
- void set_time_stamp (uint64_t time_stamp)
- void set_tow (uint32_t tow)
- void set_week (uint32_t week)

12.132.1 Detailed Description

This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
Definition at line 40 of file galileo_cnav_message.h.

12.132.2 Member Function Documentation

12.132.2.1 get_flag_CRC_test()

```
bool Galileo_Cnav_Message::get_flag_CRC_test () const  [inline]
```
Definition at line 67 of file galileo_cnav_message.h.

12.132.2.2 get_HAS_encoded_page()

```
Galileo_HAS_page Galileo_Cnav_Message::get_HAS_encoded_page () const  [inline]
```
Definition at line 62 of file galileo_cnav_message.h.

12.132.2.3 have_new_HAS_page()

```
bool Galileo_Cnav_Message::have_new_HAS_page () const  [inline]
```
Definition at line 57 of file galileo_cnav_message.h.

12.132.2.4 is_HAS_in_test_mode()

```
bool Galileo_Cnav_Message::is_HAS_in_test_mode () const  [inline]
```
Definition at line 47 of file galileo_cnav_message.h.

12.132.2.5 is_HAS_page_dummy()

```
bool Galileo_Cnav_Message::is_HAS_page_dummy () const  [inline]
```
Definition at line 52 of file galileo_cnav_message.h.

12.132.2.6 set_time_stamp()

```
void Galileo_Cnav_Message::set_time_stamp (
            uint64_t time_stamp)  [inline]
```
Definition at line 72 of file galileo_cnav_message.h.

12.132.2.7 set_tow()

```
void Galileo_Cnav_Message::set_tow (
            uint32_t tow)  [inline]
```
Definition at line 77 of file galileo_cnav_message.h.

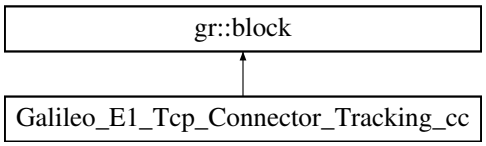
12.132.2.8 set_week()

```
void Galileo_Cnav_Message::set_week (
            uint32_t week)  [inline]
```
Definition at line 82 of file galileo_cnav_message.h.
The documentation for this class was generated from the following file:

- galileo_cnav_message.h

12.133 Galileo_E1_Tcp_Connector_Tracking_cc Class Reference

This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.
`#include <galileo_e1_tcp_connector_tracking_cc.h>`
Inheritance diagram for Galileo_E1_Tcp_Connector_Tracking_cc:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro *p_gnss_synchro)
- void **start_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)

Friends

- galileo_e1_tcp_connector_tracking_cc_sptr **galileo_e1_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float early_late_space_chips, float very_early_late_space_chips, size_t port_ch0)

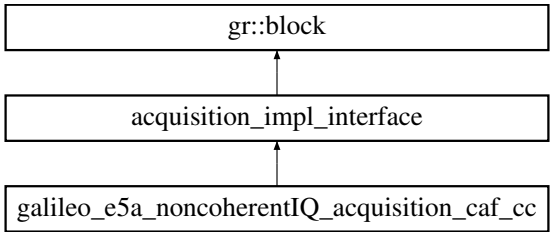
12.133.1 Detailed Description

This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.
Definition at line 63 of file galileo_e1_tcp_connector_tracking_cc.h.
The documentation for this class was generated from the following file:

- galileo_e1_tcp_connector_tracking_cc.h

12.134 galileo_e5a_noncoherentIQ_acquisition_caf_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition.
`#include <galileo_e5a_noncoherent_iq_acquisition_caf_cc.h>`
Inheritance diagram for galileo_e5a_noncoherentIQ_acquisition_caf_cc:



Public Member Functions

- ~galileo_e5a_noncoherentIQ_acquisition_caf_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- unsigned int mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code, std::complex< float > *codeQ) override

  Sets local code for PCPS acquisition algorithm.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (unsigned int channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_channel (uint32_t channel_id)=0
- virtual void set_local_code (std::complex< float > *)

Friends

- galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr **galileo_e5a_noncoherentIQ_make_acquisition_↩ caf_cc** (const Acq_Conf &conf, bool both_signal_components, int CAF_window_hz, int Zero_padding)

12.134.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Check An Open Source Galileo E1 Software Receiver, Algorithm 1, for a pseudocode description of this implementation.
Definition at line 62 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.2 Constructor & Destructor Documentation

12.134.2.1 ~galileo_e5a_noncoherentIQ_acquisition_caf_cc()

```
galileo_e5a_noncoherentIQ_acquisition_caf_cc::~galileo_e5a_noncoherentIQ_acquisition_caf_cc ()
```

Default destructor.

12.134.3 Member Function Documentation

12.134.3.1 general_work()

```
int galileo_e5a_noncoherentIQ_acquisition_caf_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.134.3.2 mag()

```
unsigned int galileo_e5a_noncoherentIQ_acquisition_caf_cc::mag () const  [inline], [override],
[virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 83 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.3.3 set_active()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_active (
            bool active)  [inline], [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 99 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.3.4 set_channel()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_channel (
            unsigned int channel)  [inline], [override]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Definition at line 113 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.3.5 set_channel_fsm()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 121 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.3.6 set_gnss_synchro()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 75 of file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

12.134.3.7 set_local_code()

```
void galileo_e5a_noncoherentIQ_acquisition_caf_cc::set_local_code (
            std::complex< float > * code,
            std::complex< float > * codeQ)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

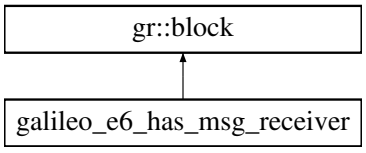
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- galileo_e5a_noncoherent_iq_acquisition_caf_cc.h

12.135 galileo_e6_has_msg_receiver Class Reference

GNU Radio block that receives asynchronous Galileo HAS message pages from the telemetry blocks, stores them in memory, and decodes HAS messages when enough data have been received. The decoded HAS message is sent to the PVT block.
`#include <galileo_e6_has_msg_receiver.h>`
Inheritance diagram for galileo_e6_has_msg_receiver:



Public Member Functions

- ~galileo_e6_has_msg_receiver ()=default

  Default destructor.
- void **set_enable_navdata_monitor** (bool enable)
- std::shared_ptr< Galileo_HAS_data > process_test_page (const pmt::pmt_t &msg)

  For testing purposes only.

Friends

- galileo_e6_has_msg_receiver_sptr **galileo_e6_has_msg_receiver_make** ()

12.135.1 Detailed Description

GNU Radio block that receives asynchronous Galileo HAS message pages from the telemetry blocks, stores them in memory, and decodes HAS messages when enough data have been received. The decoded HAS message is sent to the PVT block.
Definition at line 56 of file galileo_e6_has_msg_receiver.h.

12.135.2 Constructor & Destructor Documentation

12.135.2.1 ~galileo_e6_has_msg_receiver()

```
galileo_e6_has_msg_receiver::~galileo_e6_has_msg_receiver ()  [default]
```

Default destructor.

12.135.3 Member Function Documentation

12.135.3.1 process_test_page()

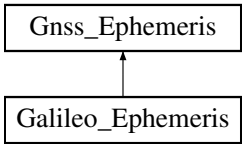
```
std::shared_ptr< Galileo_HAS_data > galileo_e6_has_msg_receiver::process_test_page (
            const pmt::pmt_t & msg)
```

For testing purposes only.
The documentation for this class was generated from the following file:

- galileo_e6_has_msg_receiver.h

12.136 Galileo_Ephemeris Class Reference

Inheritance diagram for Galileo_Ephemeris:



Public Member Functions

- double Galileo_System_Time (double week_number, double TOW)

  Galileo System Time (GST), ICD paragraph 5.1.2.
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Member Functions inherited from Gnss_Ephemeris

- double sv_clock_drift (double transmitTime)

  Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- int32_t IOD_ephemeris {}
- int32_t IOD_nav {}
- int32_t SISA {-1}

  Signal in space accuracy index (-1 if unavailable in the source data).
- int32_t E5a_HS {}

  E5a Signal Health Status.
- int32_t E5b_HS {}

  E5b Signal Health Status.
- int32_t E1B_HS {}

  E1B Signal Health Status.
- bool E5a_DVS {}

  E5a Data Validity Status.
- bool E5b_DVS {}

  E5b Data Validity Status.
- bool E1B_DVS {}

  E1B Data Validity Status.
- double BGD_E1E5a {}

  E1-E5a Broadcast Group Delay [s].
- double BGD_E1E5b {}

  E1-E5b Broadcast Group Delay [s] Navigation message carrying this record.
- Galileo_Nav_Message_Type nav_message_type {Galileo_Nav_Message_Type::Unknown}
- Galileo_Nav_Message_Source nav_message_source {Galileo_Nav_Message_Source::Unknown}

  Signal from which this navigation record was decoded.
- bool flag_all_ephemeris {}

Public Attributes inherited from Gnss_Ephemeris

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

Square root of the semi-major axis [meters^ 1/2].

- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.

- double OMEGA_0 {}

  Longitude of ascending node of orbital plane at weekly epoch [rad].

- double i_0 {}

  Inclination angle at reference time [rad].

- double omega {}

  Argument of perigee [rad].

- double OMEGAdot {}

  Rate of right ascension [rad/sec].

- double idot {}

  Rate of inclination angle [rad/sec].

- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].

- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].

- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].

- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].

- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].

- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].

- int32_t toe {}

  Ephemeris reference time [s].

- int32_t toc {}

  Clock correction data reference Time of Week [sec].

- double af0 {}

  SV clock bias correction coefficient [s].

- double af1 {}

  SV clock drift correction coefficient [s/s].

- double af2 {}

  SV clock drift rate correction coefficient [s/s^ 2].

- double satClkDrift {}

  SV clock drift.

- double dtr {}

  Relativistic clock correction term.

- int32_t WN {}

  Week number.

- int32_t tow {}

  Time of Week.

- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.

- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

- double satpos_Z {}

Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).

- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].

- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].

- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Additional Inherited Members

Protected Attributes inherited from Gnss_Ephemeris

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.136.1 Detailed Description

Definition at line 57 of file galileo_ephemeris.h.

12.136.2 Constructor & Destructor Documentation

12.136.2.1 Galileo_Ephemeris()

```
Galileo_Ephemeris::Galileo_Ephemeris ()  [inline]
```

Definition at line 60 of file galileo_ephemeris.h.

12.136.3 Member Function Documentation

12.136.3.1 Galileo_System_Time()

```
double Galileo_Ephemeris::Galileo_System_Time (
            double week_number,
            double TOW)
```

Galileo System Time (GST), ICD paragraph 5.1.2.

12.136.3.2 serialize()

```
template<class Archive>
void Galileo_Ephemeris::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Definition at line 93 of file galileo_ephemeris.h.

References Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, BGD_E1E5a, BGD_E1E5b, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::dtr, E1B_DVS, E1B_HS, E5a_DVS, E5a_HS, E5b_DVS, E5b_HS, Gnss_Ephemeris::ecc, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, Gnss_Ephemeris::M_0, nav_message_source, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdot, Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, SISA, Gnss_Ephemeris::sqrtA, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, Gnss_Ephemeris::tow, and Gnss_Ephemeris::WN.

12.136.4 Member Data Documentation

12.136.4.1 BGD_E1E5a

```
double Galileo_Ephemeris::BGD_E1E5a {}
```

E1-E5a Broadcast Group Delay [s].

Definition at line 78 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.2 BGD_E1E5b

```
double Galileo_Ephemeris::BGD_E1E5b {}
```
E1-E5b Broadcast Group Delay [s] Navigation message carrying this record.
Definition at line 79 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.3 E1B_DVS

```
bool Galileo_Ephemeris::E1B_DVS {}
```
E1B Data Validity Status.
Definition at line 77 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.4 E1B_HS

```
int32_t Galileo_Ephemeris::E1B_HS {}
```
E1B Signal Health Status.
Definition at line 74 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.5 E5a_DVS

```
bool Galileo_Ephemeris::E5a_DVS {}
```
E5a Data Validity Status.
Definition at line 75 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.6 E5a_HS

```
int32_t Galileo_Ephemeris::E5a_HS {}
```
E5a Signal Health Status.
Definition at line 72 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.7 E5b_DVS

```
bool Galileo_Ephemeris::E5b_DVS {}
```
E5b Data Validity Status.
Definition at line 76 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.8 E5b_HS

```
int32_t Galileo_Ephemeris::E5b_HS {}
```
E5b Signal Health Status.
Definition at line 73 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.9 flag_all_ephemeris

```
bool Galileo_Ephemeris::flag_all_ephemeris {}
```
Definition at line 85 of file galileo_ephemeris.h.

12.136.4.10 IOD_ephemeris

```
int32_t Galileo_Ephemeris::IOD_ephemeris {}
```
Definition at line 67 of file galileo_ephemeris.h.

12.136.4.11 IOD_nav

```
int32_t Galileo_Ephemeris::IOD_nav {}
```
Definition at line 68 of file galileo_ephemeris.h.

12.136.4.12 nav_message_source

```
Galileo_Nav_Message_Source Galileo_Ephemeris::nav_message_source {Galileo_Nav_Message_Source::↩
Unknown}
```
Signal from which this navigation record was decoded.
Definition at line 83 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().

12.136.4.13 nav_message_type

```
Galileo_Nav_Message_Type Galileo_Ephemeris::nav_message_type {Galileo_Nav_Message_Type::↩
Unknown}
```
Definition at line 81 of file galileo_ephemeris.h.

12.136.4.14 SISA

```
int32_t Galileo_Ephemeris::SISA {-1}
```
Signal in space accuracy index (-1 if unavailable in the source data).
Definition at line 71 of file galileo_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), and serialize().
The documentation for this class was generated from the following file:

- galileo_ephemeris.h

12.137 Galileo_Ephemeris_Store Class Reference

Public Types

- using Ephemeris_Map = std::map<int, Galileo_Ephemeris>

Public Member Functions

- bool insert (const Galileo_Ephemeris &ephemeris)
- const Galileo_Ephemeris * find (int prn, Galileo_Nav_Message_Type source) const
- const Ephemeris_Map & inav () const
- Ephemeris_Map & inav ()
- const Ephemeris_Map & fnav () const
- Ephemeris_Map & fnav ()
- const Ephemeris_Map & by_source (Galileo_Nav_Message_Type source) const
- Ephemeris_Map combined_view (Galileo_Nav_Message_Type preferred_source) const
- bool empty () const
- void clear ()

12.137.1 Detailed Description

Definition at line 24 of file galileo_ephemeris_store.h.

12.137.2 Member Typedef Documentation

12.137.2.1 Ephemeris_Map

```
using Galileo_Ephemeris_Store::Ephemeris_Map = std::map<int, Galileo_Ephemeris>
```
Definition at line 27 of file galileo_ephemeris_store.h.

12.137.3 Member Function Documentation

12.137.3.1 by_source()

```
const Ephemeris_Map & Galileo_Ephemeris_Store::by_source (
            Galileo_Nav_Message_Type source) const  [inline]
```
Definition at line 71 of file galileo_ephemeris_store.h.

12.137.3.2 clear()

```
void Galileo_Ephemeris_Store::clear ()  [inline]
```
Definition at line 117 of file galileo_ephemeris_store.h.

12.137.3.3 combined_view()

```
Ephemeris_Map Galileo_Ephemeris_Store::combined_view (
            Galileo_Nav_Message_Type preferred_source) const  [inline]
```
Definition at line 78 of file galileo_ephemeris_store.h.

12.137.3.4 empty()

```
bool Galileo_Ephemeris_Store::empty () const  [inline]
```
Definition at line 112 of file galileo_ephemeris_store.h.

12.137.3.5 find()

```
const Galileo_Ephemeris * Galileo_Ephemeris_Store::find (
            int prn,
            Galileo_Nav_Message_Type source) const  [inline]
```
Definition at line 40 of file galileo_ephemeris_store.h.

12.137.3.6 fnav() [1/2]

```
Ephemeris_Map & Galileo_Ephemeris_Store::fnav ()  [inline]
```
Definition at line 66 of file galileo_ephemeris_store.h.

12.137.3.7 fnav() [2/2]

```
const Ephemeris_Map & Galileo_Ephemeris_Store::fnav () const  [inline]
```
Definition at line 61 of file galileo_ephemeris_store.h.

12.137.3.8 inav() [1/2]

```
Ephemeris_Map & Galileo_Ephemeris_Store::inav ()  [inline]
```
Definition at line 56 of file galileo_ephemeris_store.h.

12.137.3.9 inav() [2/2]

```
const Ephemeris_Map & Galileo_Ephemeris_Store::inav () const  [inline]
```
Definition at line 51 of file galileo_ephemeris_store.h.

12.137.3.10 insert()

```
bool Galileo_Ephemeris_Store::insert (
            const Galileo_Ephemeris & ephemeris)  [inline]
```

Definition at line 29 of file galileo_ephemeris_store.h.
The documentation for this class was generated from the following file:

- galileo_ephemeris_store.h

12.138 Galileo_Fnav_Message Class Reference

This class handles the Galileo F/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/↩sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
`#include <galileo_fnav_message.h>`

Public Member Functions

- void **split_page** (const std::string &page_string)
- bool **have_new_ephemeris** ()
- bool **have_new_iono_and_GST** ()
- bool **have_new_utc_model** ()
- bool **have_new_almanac** ()
- Galileo_Ephemeris **get_ephemeris** () const
- Galileo_Iono **get_iono** () const
- Galileo_Utc_Model **get_utc_model** () const
- Galileo_Almanac_Helper **get_almanac** () const
- int32_t get_TOW1 () const
- int32_t get_WN1 () const
- int32_t get_TOW2 () const
- int32_t get_WN2 () const
- int32_t get_TOW3 () const
- int32_t get_WN3 () const
- int32_t get_TOW4 () const
- bool get_flag_CRC_test () const
- bool get_flag_TOW_set () const
- void set_flag_TOW_set (bool flag_tow)
- bool is_TOW1_set () const
- void set_TOW1_flag (bool flag_tow1)
- bool is_TOW2_set () const
- void set_TOW2_flag (bool flag_tow2)
- bool is_TOW3_set () const
- void set_TOW3_flag (bool flag_tow3)
- bool is_TOW4_set () const
- void set_TOW4_flag (bool flag_tow4)

12.138.1 Detailed Description

This class handles the Galileo F/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/↩sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
Definition at line 50 of file galileo_fnav_message.h.

12.138.2 Member Function Documentation

12.138.2.1 get_flag_CRC_test()

```
bool Galileo_Fnav_Message::get_flag_CRC_test () const  [inline]
```

Definition at line 100 of file galileo_fnav_message.h.

12.138.2.2 get_flag_TOW_set()

```
bool Galileo_Fnav_Message::get_flag_TOW_set () const  [inline]
```
Definition at line 105 of file galileo_fnav_message.h.

12.138.2.3 get_TOW1()

```
int32_t Galileo_Fnav_Message::get_TOW1 () const  [inline]
```
Definition at line 65 of file galileo_fnav_message.h.

12.138.2.4 get_TOW2()

```
int32_t Galileo_Fnav_Message::get_TOW2 () const  [inline]
```
Definition at line 75 of file galileo_fnav_message.h.

12.138.2.5 get_TOW3()

```
int32_t Galileo_Fnav_Message::get_TOW3 () const  [inline]
```
Definition at line 85 of file galileo_fnav_message.h.

12.138.2.6 get_TOW4()

```
int32_t Galileo_Fnav_Message::get_TOW4 () const  [inline]
```
Definition at line 95 of file galileo_fnav_message.h.

12.138.2.7 get_WN1()

```
int32_t Galileo_Fnav_Message::get_WN1 () const  [inline]
```
Definition at line 70 of file galileo_fnav_message.h.

12.138.2.8 get_WN2()

```
int32_t Galileo_Fnav_Message::get_WN2 () const  [inline]
```
Definition at line 80 of file galileo_fnav_message.h.

12.138.2.9 get_WN3()

```
int32_t Galileo_Fnav_Message::get_WN3 () const  [inline]
```
Definition at line 90 of file galileo_fnav_message.h.

12.138.2.10 is_TOW1_set()

```
bool Galileo_Fnav_Message::is_TOW1_set () const  [inline]
```
Definition at line 115 of file galileo_fnav_message.h.

12.138.2.11 is_TOW2_set()

```
bool Galileo_Fnav_Message::is_TOW2_set () const  [inline]
```
Definition at line 125 of file galileo_fnav_message.h.

12.138.2.12 is_TOW3_set()

```
bool Galileo_Fnav_Message::is_TOW3_set () const  [inline]
```
Definition at line 135 of file galileo_fnav_message.h.

12.138.2.13 is_TOW4_set()

```
bool Galileo_Fnav_Message::is_TOW4_set () const  [inline]
```
Definition at line 145 of file galileo_fnav_message.h.

12.138.2.14 set_flag_TOW_set()

```
void Galileo_Fnav_Message::set_flag_TOW_set (
            bool flag_tow)  [inline]
```

Definition at line 110 of file galileo_fnav_message.h.

12.138.2.15 set_TOW1_flag()

```
void Galileo_Fnav_Message::set_TOW1_flag (
            bool flag_tow1)  [inline]
```

Definition at line 120 of file galileo_fnav_message.h.

12.138.2.16 set_TOW2_flag()

```
void Galileo_Fnav_Message::set_TOW2_flag (
            bool flag_tow2)  [inline]
```

Definition at line 130 of file galileo_fnav_message.h.

12.138.2.17 set_TOW3_flag()

```
void Galileo_Fnav_Message::set_TOW3_flag (
            bool flag_tow3)  [inline]
```

Definition at line 140 of file galileo_fnav_message.h.

12.138.2.18 set_TOW4_flag()

```
void Galileo_Fnav_Message::set_TOW4_flag (
            bool flag_tow4)  [inline]
```

Definition at line 150 of file galileo_fnav_message.h.
The documentation for this class was generated from the following file:

- galileo_fnav_message.h

12.139 Galileo_HAS_data Class Reference

This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service Signal-In-↩ Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022. See https://www.gsc-europa.↩ eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf.
`#include <galileo_has_data.h>`

Public Member Functions

- std::vector< std::string > get_signals_in_mask (uint8_t nsys) const

  Get a vector of Nsys std::string with signals in mask for system nsys, with 0 <= nsys < Nsys.
- std::vector< std::string > get_signals_in_mask (const std::string &system) const

  Get a vector of Nsys std::string with signals in mask for system ("GPS"/"Galileo").
- std::vector< std::string > get_systems_string () const

  Get Nsys system name strings.
- std::vector< std::string > get_systems_subset_string () const

  Get Nsat system name strings present in clock corrections subset.
- std::vector< std::vector< float > > get_code_bias_m () const

  Get Nsat x Ncodes code biases in [m].
- std::vector< std::vector< float > > get_phase_bias_cycle () const

  Get Nsat x Nphases phase biases in [cycles].
- std::vector< std::vector< float > > get_delta_clock_subset_correction_m () const

  Get Nsys_sub vectors with Nsat_sub delta clock C0 corrections in [m].
- std::vector< float > get_delta_radial_m () const

Get Nsat delta radial corrections in [m].

- std::vector< float > get_delta_radial_m (uint8_t nsys) const

  Get delta radial corrections in [m] for system nsys, with 0 <= nsys < Nsys.
- std::vector< float > get_delta_in_track_m () const

  Get Nsat delta in-track corrections in [m].
- std::vector< float > get_delta_in_track_m (uint8_t nsys) const

  Get delta in-track corrections in [m] for system nsys, with 0 <= nsys < Nsys.
- std::vector< float > get_delta_cross_track_m () const

  Get Nsat delta cross-track corrections in [m].
- std::vector< float > get_delta_cross_track_m (uint8_t nsys) const

  Get delta cross-track corrections in [m] for system nsys, with 0 <= nsys < Nsys.
- std::vector< float > get_delta_clock_correction_m () const

  Get Nsat delta clock C0 corrections in [m].
- std::vector< float > get_delta_clock_correction_m (uint8_t nsys) const

  Get delta clock C0 corrections in [m] for system nsys, with 0 <= nsys < Nsys.
- std::vector< float > get_delta_clock_subset_correction_m (uint8_t nsys) const

  Get delta clock C0 subset corrections in [m] for system nsys, with 0 <= nsys < Nsys.
- std::vector< int > get_PRNs_in_mask (uint8_t nsys) const

  Get PRNs in mask for system nsys, with 0 <= nsys < Nsys.
- std::vector< int > get_PRNs_in_mask (const std::string &system) const

  Get PRNs in mask for system ("GPS"/"Galileo").
- std::vector< int > get_PRNs_in_submask (uint8_t nsys) const

  Get PRNs in submask for system nsys, with 0 <= nsys < Nsys.
- std::vector< uint16_t > get_gnss_iod (uint8_t nsys) const

  Get GNSS IODs for for system nsys, with 0 <= nsys < Nsys.
- std::vector< uint8_t > get_num_satellites () const

  Get Nsys number of satellites.
- std::vector< uint8_t > get_num_subset_satellites () const

  Get Nsys_sub number of satellites.
- float get_code_bias_m (const std::string &signal, int PRN) const

  Get code bias in [m] for a given signal and PRN satellite.
- float get_phase_bias_cycle (const std::string &signal, int PRN) const

  Get phase bias in [cycles] for a given signal and PRN satellite.
- float get_delta_radial_m (const std::string &system, int prn) const

  Get orbital radial correction in [m] for a given system ("GPS"/"Galileo") and PRN.
- float get_delta_in_track_m (const std::string &system, int prn) const

  Get orbital in_track correction in [m] for a given system ("GPS"/"Galileo") and PRN.
- float get_delta_cross_track_m (const std::string &system, int prn) const

  Get orbital cross_track correction in [m] for a given system ("GPS"/"Galileo") and PRN.
- float get_clock_correction_mult_m (const std::string &system, int prn) const

  Get clock correction in [m], already multiplied by its Delta Clock Multiplier, for a given system ("GPS"/"Galileo") and PRN.
- float get_clock_subset_correction_mult_m (const std::string &system, int prn) const

  Get clock correction subset in [m], already multiplied by its Delta Clock Multiplier.
- uint16_t get_nsat () const

  Get total number of satellites with corrections.
- uint16_t get_nsat_sub () const

  Get number of satellites in clock subset corrections.
- uint16_t get_validity_interval_s (uint8_t validity_interval_index) const

  Get validity interval in [s] from the validity_interval_index.
- uint32_t get_time_of_message_s () const

Get Time of Message, in [s] of week. See HAS SIS ICD 1.0 Section 7.7.

- uint16_t get_gnss_iod (const std::string &system, int prn) const

  Get GNSS IOD from a given system ("GPS"/"Galileo") and PRN.
- uint8_t get_gnss_id (int nsat) const

  Get GNSS ID from the nsat satellite.
- uint8_t get_phase_discontinuity_indicator (const std::string &signal, int PRN) const

  Get PDI for a given signal and PRN satellite.

Static Public Member Functions

- static bool has_validity_interval_at_tow (uint32_t valid_until, uint32_t tow)

  Check whether a validity end time is active at the given TOW.
- static bool is_orbit_radial_unavailable (float value)

  Check HAS radial correction unavailable sentinel.
- static bool is_orbit_along_cross_unavailable (float value)

  Check HAS along/cross-track correction unavailable sentinel.
- static bool is_clock_unavailable (float value)

  Check HAS clock correction unavailable sentinel.
- static bool is_clock_do_not_use (float value)

  Check HAS clock correction do-not-use sentinel.
- static bool is_code_bias_unavailable (float value)

  Check HAS code bias unavailable sentinel.
- static bool is_phase_bias_unavailable (float value)

  Check HAS phase bias unavailable sentinel.

Public Attributes

- std::vector< uint8_t > gnss_id_mask

  GNSS ID. See HAS SIS ICD 1.0 Section 5.2.1.1.
- std::vector< uint64_t > satellite_mask

  SatM - Satellite Mask. See HAS SIS ICD 1.0 Section 5.2.1.2.
- std::vector< uint16_t > signal_mask

  SigM - Signal Mask. See HAS SIS ICD 1.0 Section 5.2.1.3.
- std::vector< bool > cell_mask_availability_flag

  CMAF - Cell Mask Availability Flag. See HAS SIS ICD 1.0 Section 5.2.1.4.
- std::vector< std::vector< std::vector< bool > > > cell_mask

  CM - Cell Mask. See HAS SIS ICD 1.0 Section 5.2.1.5.
- std::vector< uint8_t > nav_message

  NM - Navigation Message Index. See HAS SIS ICD 1.0 Section 5.2.1.6.
- std::vector< uint16_t > gnss_iod

  IODref - Reference Issue of Data. See HAS SIS ICD 1.0 Table 26.
- std::vector< int16_t > delta_radial

  DR - Delta Radial Correction. See HAS SIS ICD 1.0 Table 25.
- std::vector< int16_t > delta_in_track

  DIT - Delta In-Track Correction. See HAS SIS ICD 1.0 Table 25.
- std::vector< int16_t > delta_cross_track

  DCT - Delta Cross Correction. See HAS SIS ICD 1.0 Table 25.
- std::vector< uint8_t > delta_clock_multiplier

  DCM - Delta Clock Multipliers. See HAS SIS ICD 1.0 Section 5.2.3.1.
- std::vector< int16_t > delta_clock_correction

  DCC - Delta Clock Corrections. See HAS SIS ICD 1.0 Section 5.2.3.2.
- std::vector< uint8_t > gnss_id_clock_subset

GNSS ID. Specific GNSS to which the corrections refer. See HAS SIS ICD 1.0 Section 5.2.1.1.

- std::vector< uint8_t > delta_clock_multiplier_clock_subset

  DCM. Multiplier for all Delta Clock corrections. See HAS SIS ICD 1.0 Section 5.2.3.1.
- std::vector< uint64_t > satellite_submask

  SatMsub - Satellite Subset Mask. See HAS SIS ICD 1.0 Section 5.2.4.1.
- std::vector< std::vector< int16_t > > delta_clock_correction_clock_subset

  DCCsub - Delta Clock Subset Corrections. See HAS SIS ICD 1.0 Section 5.2.4.1.
- std::vector< std::vector< int16_t > > code_bias

  CB - Code bias for the m-th signal of the n-th SV. See HAS SIS ICD 1.0 Section 5.2.5.
- std::vector< std::vector< int16_t > > phase_bias

  PB - Phase bias for the m-th signal of the n-th SV. See HAS SIS ICD 1.0 Section 5.2.6.
- std::vector< std::vector< uint8_t > > phase_discontinuity_indicator

  PDI - Phase Discontinuity Indicator. See HAS SIS ICD 1.0 Section 5.2.6.
- uint32_t week {GALILEO_HAS_INVALID_WEEK}

  Galileo week number.
- uint32_t tow {GALILEO_HAS_INVALID_TOW}

  Time of Week.
- mt1_header header

  MT1 Header parameters. See HAS SIS ICD 1.0 Section 5.1.1.
- uint8_t has_status

  HASS - HAS Status (from HAS page header). See HAS SIS ICD 1.0 Section 3.1.1.
- uint8_t message_id

  MID - Message ID (from HAS page header). See HAS SIS ICD 1.0 Section 3.1.
- uint8_t Nsys

  Number of GNSS for which corrections are provided. See HAS SIS ICD 1.0 Section 5.2.1.
- uint8_t Nsys_sub

  Number of GNSS for which corrections are provided in clock subset corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
- uint8_t validity_interval_index_orbit_corrections

  VI - Validity Interval Index for Orbit corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
- uint8_t validity_interval_index_clock_fullset_corrections

  VI - Validity Interval Index for Clock full-set corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
- uint8_t validity_interval_index_clock_subset_corrections

  VI - Validity Interval Index for Clock subset corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
- uint8_t validity_interval_index_code_bias_corrections

  VI - Validity Interval Index for Code bias. See HAS SIS ICD 1.0 Section 5.2.2.1.
- uint8_t validity_interval_index_phase_bias_corrections

  VI - Validity Interval Index for Phase bias. See HAS SIS ICD 1.0 Section 5.2.2.1.

12.139.1 Detailed Description

This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service Signal-In-↩ Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022. See https://www.gsc-europa.↩ eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf.
Definition at line 51 of file galileo_has_data.h.

12.139.2 Member Function Documentation

12.139.2.1 get_clock_correction_mult_m()

```
float Galileo_HAS_data::get_clock_correction_mult_m (
            const std::string & system,
            int prn) const
```

Get clock correction in [m], already multiplied by its Delta Clock Multiplier, for a given system ("GPS"/"Galileo") and PRN.

12.139.2.2 get_clock_subset_correction_mult_m()

```
float Galileo_HAS_data::get_clock_subset_correction_mult_m (
            const std::string & system,
            int prn) const
```

Get clock correction subset in [m], already multiplied by its Delta Clock Multiplier.

12.139.2.3 get_code_bias_m() [1/2]

```
std::vector< std::vector< float > > Galileo_HAS_data::get_code_bias_m () const
```

Get Nsat x Ncodes code biases in [m].

12.139.2.4 get_code_bias_m() [2/2]

```
float Galileo_HAS_data::get_code_bias_m (
            const std::string & signal,
            int PRN) const
```

Get code bias in [m] for a given signal and PRN satellite.

12.139.2.5 get_delta_clock_correction_m() [1/2]

```
std::vector< float > Galileo_HAS_data::get_delta_clock_correction_m () const
```

Get Nsat delta clock C0 corrections in [m].

12.139.2.6 get_delta_clock_correction_m() [2/2]

```
std::vector< float > Galileo_HAS_data::get_delta_clock_correction_m (
            uint8_t nsys) const
```

Get delta clock C0 corrections in [m] for system nsys, with 0 <= nsys < Nsys.

12.139.2.7 get_delta_clock_subset_correction_m() [1/2]

```
std::vector< std::vector< float > > Galileo_HAS_data::get_delta_clock_subset_correction_m ()
const
```

Get Nsys_sub vectors with Nsat_sub delta clock C0 corrections in [m].

12.139.2.8 get_delta_clock_subset_correction_m() [2/2]

```
std::vector< float > Galileo_HAS_data::get_delta_clock_subset_correction_m (
            uint8_t nsys) const
```

Get delta clock C0 subset corrections in [m] for system nsys, with 0 <= nsys < Nsys.

12.139.2.9 get_delta_cross_track_m() [1/3]

```
std::vector< float > Galileo_HAS_data::get_delta_cross_track_m () const
```

Get Nsat delta cross-track corrections in [m].

12.139.2.10 get_delta_cross_track_m() [2/3]

```
float Galileo_HAS_data::get_delta_cross_track_m (
            const std::string & system,
            int prn) const
```

Get orbital cross_track correction in [m] for a given system ("GPS"/"Galileo") and PRN.

12.139.2.11 get_delta_cross_track_m() [3/3]

```
std::vector< float > Galileo_HAS_data::get_delta_cross_track_m (
            uint8_t nsys) const
```

Get delta cross-track corrections in [m] for system nsys, with 0 <= nsys < Nsys.

12.139.2.12 get_delta_in_track_m() [1/3]

```
std::vector< float > Galileo_HAS_data::get_delta_in_track_m () const
```

Get Nsat delta in-track corrections in [m].

12.139.2.13 get_delta_in_track_m() [2/3]

```
float Galileo_HAS_data::get_delta_in_track_m (
            const std::string & system,
            int prn) const
```

Get orbital in_track correction in [m] for a given system ("GPS"/"Galileo") and PRN.

12.139.2.14 get_delta_in_track_m() [3/3]

```
std::vector< float > Galileo_HAS_data::get_delta_in_track_m (
            uint8_t nsys) const
```

Get delta in-track corrections in [m] for system nsys, with 0 <= nsys < Nsys.

12.139.2.15 get_delta_radial_m() [1/3]

```
std::vector< float > Galileo_HAS_data::get_delta_radial_m () const
```

Get Nsat delta radial corrections in [m].

12.139.2.16 get_delta_radial_m() [2/3]

```
float Galileo_HAS_data::get_delta_radial_m (
            const std::string & system,
            int prn) const
```

Get orbital radial correction in [m] for a given system ("GPS"/"Galileo") and PRN.

12.139.2.17 get_delta_radial_m() [3/3]

```
std::vector< float > Galileo_HAS_data::get_delta_radial_m (
            uint8_t nsys) const
```

Get delta radial corrections in [m] for system nsys, with 0 <= nsys < Nsys.

12.139.2.18 get_gnss_id()

```
uint8_t Galileo_HAS_data::get_gnss_id (
            int nsat) const
```

Get GNSS ID from the nsat satellite.

12.139.2.19 get_gnss_iod() [1/2]

```
uint16_t Galileo_HAS_data::get_gnss_iod (
            const std::string & system,
            int prn) const
```

Get GNSS IOD from a given system ("GPS"/"Galileo") and PRN.

12.139.2.20 get_gnss_iod() [2/2]

```
std::vector< uint16_t > Galileo_HAS_data::get_gnss_iod (
            uint8_t nsys) const
```

Get GNSS IODs for for system nsys, with 0 <= nsys < Nsys.

12.139.2.21 get_nsat()

```
uint16_t Galileo_HAS_data::get_nsat () const
```

Get total number of satellites with corrections.

12.139.2.22 get_nsat_sub()

```
uint16_t Galileo_HAS_data::get_nsat_sub () const
```
Get number of satellites in clock subset corrections.

12.139.2.23 get_num_satellites()

```
std::vector< uint8_t > Galileo_HAS_data::get_num_satellites () const
```
Get Nsys number of satellites.

12.139.2.24 get_num_subset_satellites()

```
std::vector< uint8_t > Galileo_HAS_data::get_num_subset_satellites () const
```
Get Nsys_sub number of satellites.

12.139.2.25 get_phase_bias_cycle() [1/2]

```
std::vector< std::vector< float > > Galileo_HAS_data::get_phase_bias_cycle () const
```
Get Nsat x Nphases phase biases in [cycles].

12.139.2.26 get_phase_bias_cycle() [2/2]

```
float Galileo_HAS_data::get_phase_bias_cycle (
            const std::string & signal,
            int PRN) const
```
Get phase bias in [cycles] for a given signal and PRN satellite.

12.139.2.27 get_phase_discontinuity_indicator()

```
uint8_t Galileo_HAS_data::get_phase_discontinuity_indicator (
            const std::string & signal,
            int PRN) const
```
Get PDI for a given signal and PRN satellite.

12.139.2.28 get_PRNs_in_mask() [1/2]

```
std::vector< int > Galileo_HAS_data::get_PRNs_in_mask (
            const std::string & system) const
```
Get PRNs in mask for system ("GPS"/"Galileo").

12.139.2.29 get_PRNs_in_mask() [2/2]

```
std::vector< int > Galileo_HAS_data::get_PRNs_in_mask (
            uint8_t nsys) const
```
Get PRNs in mask for system nsys, with 0 <= nsys < Nsys.

12.139.2.30 get_PRNs_in_submask()

```
std::vector< int > Galileo_HAS_data::get_PRNs_in_submask (
            uint8_t nsys) const
```
Get PRNs in submask for system nsys, with 0 <= nsys < Nsys.

12.139.2.31 get_signals_in_mask() [1/2]

```
std::vector< std::string > Galileo_HAS_data::get_signals_in_mask (
            const std::string & system) const
```
Get a vector of Nsys std::string with signals in mask for system ("GPS"/"Galileo").

12.139.2.32 get_signals_in_mask() [2/2]

```
std::vector< std::string > Galileo_HAS_data::get_signals_in_mask (
            uint8_t nsys) const
```

Get a vector of Nsys std::string with signals in mask for system nsys, with 0 <= nsys < Nsys.

12.139.2.33 get_systems_string()

```
std::vector< std::string > Galileo_HAS_data::get_systems_string () const
```

Get Nsys system name strings.

12.139.2.34 get_systems_subset_string()

```
std::vector< std::string > Galileo_HAS_data::get_systems_subset_string () const
```

Get Nsat system name strings present in clock corrections subset.

12.139.2.35 get_time_of_message_s()

```
uint32_t Galileo_HAS_data::get_time_of_message_s () const
```

Get Time of Message, in [s] of week. See HAS SIS ICD 1.0 Section 7.7.

12.139.2.36 get_validity_interval_s()

```
uint16_t Galileo_HAS_data::get_validity_interval_s (
            uint8_t validity_interval_index) const
```

Get validity interval in [s] from the validity_interval_index.

12.139.2.37 has_validity_interval_at_tow()

```
bool Galileo_HAS_data::has_validity_interval_at_tow (
            uint32_t valid_until,
            uint32_t tow)  [static]
```

Check whether a validity end time is active at the given TOW.
References tow.

12.139.2.38 is_clock_do_not_use()

```
bool Galileo_HAS_data::is_clock_do_not_use (
            float value)  [static]
```

Check HAS clock correction do-not-use sentinel.

12.139.2.39 is_clock_unavailable()

```
bool Galileo_HAS_data::is_clock_unavailable (
            float value)  [static]
```

Check HAS clock correction unavailable sentinel.

12.139.2.40 is_code_bias_unavailable()

```
bool Galileo_HAS_data::is_code_bias_unavailable (
            float value)  [static]
```

Check HAS code bias unavailable sentinel.

12.139.2.41 is_orbit_along_cross_unavailable()

```
bool Galileo_HAS_data::is_orbit_along_cross_unavailable (
            float value)  [static]
```

Check HAS along/cross-track correction unavailable sentinel.

12.139.2.42 is_orbit_radial_unavailable()

```
bool Galileo_HAS_data::is_orbit_radial_unavailable (
            float value)  [static]
```

Check HAS radial correction unavailable sentinel.

12.139.2.43 is_phase_bias_unavailable()

```
bool Galileo_HAS_data::is_phase_bias_unavailable (
            float value)  [static]
```

Check HAS phase bias unavailable sentinel.

12.139.3 Member Data Documentation

12.139.3.1 cell_mask

```
std::vector<std::vector<std::vector<bool> > > Galileo_HAS_data::cell_mask
```

CM - Cell Mask. See HAS SIS ICD 1.0 Section 5.2.1.5.
Definition at line 105 of file galileo_has_data.h.

12.139.3.2 cell_mask_availability_flag

```
std::vector<bool> Galileo_HAS_data::cell_mask_availability_flag
```

CMAF - Cell Mask Availability Flag. See HAS SIS ICD 1.0 Section 5.2.1.4.
Definition at line 104 of file galileo_has_data.h.

12.139.3.3 code_bias

```
std::vector<std::vector<int16_t> > Galileo_HAS_data::code_bias
```

CB - Code bias for the m-th signal of the n-th SV. See HAS SIS ICD 1.0 Section 5.2.5.
Definition at line 125 of file galileo_has_data.h.

12.139.3.4 delta_clock_correction

```
std::vector<int16_t> Galileo_HAS_data::delta_clock_correction
```

DCC - Delta Clock Corrections. See HAS SIS ICD 1.0 Section 5.2.3.2.
Definition at line 116 of file galileo_has_data.h.

12.139.3.5 delta_clock_correction_clock_subset

```
std::vector<std::vector<int16_t> > Galileo_HAS_data::delta_clock_correction_clock_subset
```

DCCsub - Delta Clock Subset Corrections. See HAS SIS ICD 1.0 Section 5.2.4.1.
Definition at line 122 of file galileo_has_data.h.

12.139.3.6 delta_clock_multiplier

```
std::vector<uint8_t> Galileo_HAS_data::delta_clock_multiplier
```

DCM - Delta Clock Multipliers. See HAS SIS ICD 1.0 Section 5.2.3.1.
Definition at line 115 of file galileo_has_data.h.

12.139.3.7 delta_clock_multiplier_clock_subset

```
std::vector<uint8_t> Galileo_HAS_data::delta_clock_multiplier_clock_subset
```

DCM. Multiplier for all Delta Clock corrections. See HAS SIS ICD 1.0 Section 5.2.3.1.
Definition at line 120 of file galileo_has_data.h.

12.139.3.8 delta_cross_track

```
std::vector<int16_t> Galileo_HAS_data::delta_cross_track
```

DCT - Delta Cross Correction. See HAS SIS ICD 1.0 Table 25.
Definition at line 112 of file galileo_has_data.h.

12.139.3.9 delta_in_track

`std::vector<int16_t> Galileo_HAS_data::delta_in_track`
DIT - Delta In-Track Correction. See HAS SIS ICD 1.0 Table 25.
Definition at line 111 of file galileo_has_data.h.

12.139.3.10 delta_radial

`std::vector<int16_t> Galileo_HAS_data::delta_radial`
DR - Delta Radial Correction. See HAS SIS ICD 1.0 Table 25.
Definition at line 110 of file galileo_has_data.h.

12.139.3.11 gnss_id_clock_subset

`std::vector<uint8_t> Galileo_HAS_data::gnss_id_clock_subset`
GNSS ID. Specific GNSS to which the corrections refer. See HAS SIS ICD 1.0 Section 5.2.1.1.
Definition at line 119 of file galileo_has_data.h.

12.139.3.12 gnss_id_mask

`std::vector<uint8_t> Galileo_HAS_data::gnss_id_mask`
GNSS ID. See HAS SIS ICD 1.0 Section 5.2.1.1.
Definition at line 101 of file galileo_has_data.h.

12.139.3.13 gnss_iod

`std::vector<uint16_t> Galileo_HAS_data::gnss_iod`
IODref - Reference Issue of Data. See HAS SIS ICD 1.0 Table 26.
Definition at line 109 of file galileo_has_data.h.

12.139.3.14 has_status

`uint8_t Galileo_HAS_data::has_status`
HASS - HAS Status (from HAS page header). See HAS SIS ICD 1.0 Section 3.1.1.
Definition at line 135 of file galileo_has_data.h.

12.139.3.15 header

`mt1_header Galileo_HAS_data::header`
MT1 Header parameters. See HAS SIS ICD 1.0 Section 5.1.1.
Definition at line 134 of file galileo_has_data.h.

12.139.3.16 message_id

`uint8_t Galileo_HAS_data::message_id`
MID - Message ID (from HAS page header). See HAS SIS ICD 1.0 Section 3.1.
Definition at line 136 of file galileo_has_data.h.

12.139.3.17 nav_message

`std::vector<uint8_t> Galileo_HAS_data::nav_message`
NM - Navigation Message Index. See HAS SIS ICD 1.0 Section 5.2.1.6.
Definition at line 106 of file galileo_has_data.h.

12.139.3.18 Nsys

`uint8_t Galileo_HAS_data::Nsys`
Number of GNSS for which corrections are provided. See HAS SIS ICD 1.0 Section 5.2.1.
Definition at line 138 of file galileo_has_data.h.

12.139.3.19 Nsys_sub

`uint8_t Galileo_HAS_data::Nsys_sub`
Number of GNSS for which corrections are provided in clock subset corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 139 of file galileo_has_data.h.

12.139.3.20 phase_bias

`std::vector<std::vector<int16_t> > Galileo_HAS_data::phase_bias`
PB - Phase bias for the m-th signal of the n-th SV. See HAS SIS ICD 1.0 Section 5.2.6.
Definition at line 128 of file galileo_has_data.h.

12.139.3.21 phase_discontinuity_indicator

`std::vector<std::vector<uint8_t> > Galileo_HAS_data::phase_discontinuity_indicator`
PDI - Phase Discontinuity Indicator. See HAS SIS ICD 1.0 Section 5.2.6.
Definition at line 129 of file galileo_has_data.h.

12.139.3.22 satellite_mask

`std::vector<uint64_t> Galileo_HAS_data::satellite_mask`
SatM - Satellite Mask. See HAS SIS ICD 1.0 Section 5.2.1.2.
Definition at line 102 of file galileo_has_data.h.

12.139.3.23 satellite_submask

`std::vector<uint64_t> Galileo_HAS_data::satellite_submask`
SatMsub - Satellite Subset Mask. See HAS SIS ICD 1.0 Section 5.2.4.1.
Definition at line 121 of file galileo_has_data.h.

12.139.3.24 signal_mask

`std::vector<uint16_t> Galileo_HAS_data::signal_mask`
SigM - Signal Mask. See HAS SIS ICD 1.0 Section 5.2.1.3.
Definition at line 103 of file galileo_has_data.h.

12.139.3.25 tow

`uint32_t Galileo_HAS_data::tow {GALILEO_HAS_INVALID_TOW}`
Time of Week.
Definition at line 132 of file galileo_has_data.h.
Referenced by has_validity_interval_at_tow().

12.139.3.26 validity_interval_index_clock_fullset_corrections

`uint8_t Galileo_HAS_data::validity_interval_index_clock_fullset_corrections`
VI - Validity Interval Index for Clock full-set corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 142 of file galileo_has_data.h.

12.139.3.27 validity_interval_index_clock_subset_corrections

`uint8_t Galileo_HAS_data::validity_interval_index_clock_subset_corrections`
VI - Validity Interval Index for Clock subset corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 143 of file galileo_has_data.h.

12.139.3.28 validity_interval_index_code_bias_corrections

`uint8_t Galileo_HAS_data::validity_interval_index_code_bias_corrections`
VI - Validity Interval Index for Code bias. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 144 of file galileo_has_data.h.

12.139.3.29 validity_interval_index_orbit_corrections

`uint8_t Galileo_HAS_data::validity_interval_index_orbit_corrections`
VI - Validity Interval Index for Orbit corrections. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 141 of file galileo_has_data.h.

12.139.3.30 validity_interval_index_phase_bias_corrections

`uint8_t Galileo_HAS_data::validity_interval_index_phase_bias_corrections`
VI - Validity Interval Index for Phase bias. See HAS SIS ICD 1.0 Section 5.2.2.1.
Definition at line 145 of file galileo_has_data.h.

12.139.3.31 week

`uint32_t Galileo_HAS_data::week {GALILEO_HAS_INVALID_WEEK}`
Galileo week number.
Definition at line 131 of file galileo_has_data.h.
The documentation for this class was generated from the following file:

- galileo_has_data.h

12.140 Galileo_HAS_page Class Reference

This class is a storage for Galileo HAS message page, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
`#include <galileo_has_page.h>`

Public Attributes

- std::string has_message_string

  HAS message content.
- uint64_t time_stamp {}

  HAS page time stamp, in [s].
- uint32_t week {GALILEO_HAS_INVALID_WEEK}

  HAS page Galileo week number.
- uint32_t tow {GALILEO_HAS_INVALID_TOW}

  HAS page time of week, in [s].
- uint8_t has_status {}

  HAS status.
- uint8_t reserved {}

  HAS reserved field.
- uint8_t message_type {}

  HAS message type (MT).
- uint8_t message_id {}

  HAS message ID (MID).
- uint8_t message_size {}

  HAS message size (MS).
- uint8_t message_page_id {}

  HAS message page ID (PID).

12.140.1 Detailed Description

This class is a storage for Galileo HAS message page, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.
Definition at line 36 of file galileo_has_page.h.

12.140.2 Member Data Documentation

12.140.2.1 has_message_string

`std::string Galileo_HAS_page::has_message_string`
HAS message content.
Definition at line 41 of file galileo_has_page.h.

12.140.2.2 has_status

`uint8_t Galileo_HAS_page::has_status {}`
HAS status.
Definition at line 47 of file galileo_has_page.h.

12.140.2.3 message_id

`uint8_t Galileo_HAS_page::message_id {}`
HAS message ID (MID).
Definition at line 50 of file galileo_has_page.h.

12.140.2.4 message_page_id

`uint8_t Galileo_HAS_page::message_page_id {}`
HAS message page ID (PID).
Definition at line 52 of file galileo_has_page.h.

12.140.2.5 message_size

`uint8_t Galileo_HAS_page::message_size {}`
HAS message size (MS).
Definition at line 51 of file galileo_has_page.h.

12.140.2.6 message_type

`uint8_t Galileo_HAS_page::message_type {}`
HAS message type (MT).
Definition at line 49 of file galileo_has_page.h.

12.140.2.7 reserved

`uint8_t Galileo_HAS_page::reserved {}`
HAS reserved field.
Definition at line 48 of file galileo_has_page.h.

12.140.2.8 time_stamp

`uint64_t Galileo_HAS_page::time_stamp {}`
HAS page time stamp, in [s].
Definition at line 42 of file galileo_has_page.h.

12.140.2.9 tow

`uint32_t Galileo_HAS_page::tow {GALILEO_HAS_INVALID_TOW}`
HAS page time of week, in [s].
Definition at line 44 of file galileo_has_page.h.

12.140.2.10 week

`uint32_t Galileo_HAS_page::week {GALILEO_HAS_INVALID_WEEK}`
HAS page Galileo week number.
Definition at line 43 of file galileo_has_page.h.
The documentation for this class was generated from the following file:

- galileo_has_page.h

12.141 Galileo_Inav_Message Class Reference

This class handles the Galileo I/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/↩sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
`#include <galileo_inav_message.h>`

Public Member Functions

- void **split_page** (std::string page_string, int32_t flag_even_word)
- bool **have_new_ephemeris** ()
- bool **have_new_iono_and_GST** ()
- bool **have_new_utc_model** ()
- bool **have_new_almanac** ()
- bool **have_new_reduced_ced** ()
- bool **have_new_ism** ()
- bool **have_new_nma** ()
- Galileo_Ephemeris **get_ephemeris** () const
- Galileo_Iono **get_iono** () const
- Galileo_Utc_Model **get_utc_model** () const
- Galileo_Almanac_Helper **get_almanac** () const
- Galileo_Reduced_CED **get_reduced_ced** () const
- Galileo_ISM **get_galileo_ism** () const
- OSNMA_msg **get_osnma_msg** ()
- std::string **get_osnma_adkd_4_nav_bits** ()
- void **reset_osnma_nav_bits_adkd4** ()
- std::string **get_osnma_adkd_0_12_nav_bits** ()
- void **reset_osnma_nav_bits_adkd0_12** ()
- bool get_flag_CRC_test () const
- bool is_alert_page () const
- bool get_flag_TOW_set () const
- void set_flag_TOW_set (bool flag_tow)
- int32_t get_Galileo_week () const
- int32_t get_TOW5 () const
- int32_t get_TOW6 () const
- bool is_TOW5_set () const
- void set_TOW5_flag (bool flag_tow5)
- bool is_TOW6_set () const
- void set_TOW6_flag (bool flag_tow6)
- int32_t get_TOW0 () const
- bool is_TOW0_set () const
- void set_TOW0_flag (bool flag_tow0)
- bool get_flag_GGTO () const
- double get_A0G () const
- double get_A1G () const
- double get_t0G () const
- double get_WN0G () const
- void init_PRN (uint32_t prn)
- void enable_reed_solomon ()

12.141.1 Detailed Description

This class handles the Galileo I/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.
Definition at line 65 of file galileo_inav_message.h.

12.141.2 Member Function Documentation

12.141.2.1 enable_reed_solomon()

```
void Galileo_Inav_Message::enable_reed_solomon ()  [inline]
```
Definition at line 285 of file galileo_inav_message.h.

12.141.2.2 get_A0G()

```
double Galileo_Inav_Message::get_A0G () const  [inline]
```
Definition at line 242 of file galileo_inav_message.h.

12.141.2.3 get_A1G()

```
double Galileo_Inav_Message::get_A1G () const  [inline]
```
Definition at line 247 of file galileo_inav_message.h.

12.141.2.4 get_flag_CRC_test()

```
bool Galileo_Inav_Message::get_flag_CRC_test () const  [inline]
```
Definition at line 167 of file galileo_inav_message.h.

12.141.2.5 get_flag_GGTO()

```
bool Galileo_Inav_Message::get_flag_GGTO () const  [inline]
```
Definition at line 237 of file galileo_inav_message.h.

12.141.2.6 get_flag_TOW_set()

```
bool Galileo_Inav_Message::get_flag_TOW_set () const  [inline]
```
Definition at line 177 of file galileo_inav_message.h.

12.141.2.7 get_Galileo_week()

```
int32_t Galileo_Inav_Message::get_Galileo_week () const  [inline]
```
Definition at line 187 of file galileo_inav_message.h.

12.141.2.8 get_t0G()

```
double Galileo_Inav_Message::get_t0G () const  [inline]
```
Definition at line 252 of file galileo_inav_message.h.

12.141.2.9 get_TOW0()

```
int32_t Galileo_Inav_Message::get_TOW0 () const  [inline]
```
Definition at line 222 of file galileo_inav_message.h.

12.141.2.10 get_TOW5()

```
int32_t Galileo_Inav_Message::get_TOW5 () const  [inline]
```
Definition at line 192 of file galileo_inav_message.h.

12.141.2.11 get_TOW6()

```
int32_t Galileo_Inav_Message::get_TOW6 () const  [inline]
```
Definition at line 197 of file galileo_inav_message.h.

12.141.2.12 get_WN0G()

```
double Galileo_Inav_Message::get_WN0G () const  [inline]
```
Definition at line 257 of file galileo_inav_message.h.

12.141.2.13 init_PRN()

```
void Galileo_Inav_Message::init_PRN (
            uint32_t prn)  [inline]
```
Definition at line 265 of file galileo_inav_message.h.

12.141.2.14 is_alert_page()

```
bool Galileo_Inav_Message::is_alert_page () const  [inline]
```
Definition at line 172 of file galileo_inav_message.h.

12.141.2.15 is_TOW0_set()

```
bool Galileo_Inav_Message::is_TOW0_set () const  [inline]
```
Definition at line 227 of file galileo_inav_message.h.

12.141.2.16 is_TOW5_set()

```
bool Galileo_Inav_Message::is_TOW5_set () const  [inline]
```
Definition at line 202 of file galileo_inav_message.h.

12.141.2.17 is_TOW6_set()

```
bool Galileo_Inav_Message::is_TOW6_set () const  [inline]
```
Definition at line 212 of file galileo_inav_message.h.

12.141.2.18 set_flag_TOW_set()

```
void Galileo_Inav_Message::set_flag_TOW_set (
            bool flag_tow)  [inline]
```
Definition at line 182 of file galileo_inav_message.h.

12.141.2.19 set_TOW0_flag()

```
void Galileo_Inav_Message::set_TOW0_flag (
            bool flag_tow0)  [inline]
```
Definition at line 232 of file galileo_inav_message.h.

12.141.2.20 set_TOW5_flag()

```
void Galileo_Inav_Message::set_TOW5_flag (
            bool flag_tow5)  [inline]
```
Definition at line 207 of file galileo_inav_message.h.

12.141.2.21 set_TOW6_flag()

```
void Galileo_Inav_Message::set_TOW6_flag (
            bool flag_tow6)  [inline]
```
Definition at line 217 of file galileo_inav_message.h.

The documentation for this class was generated from the following file:

- galileo_inav_message.h

12.142 Galileo_Iono Class Reference

This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6.
`#include <galileo_iono.h>`

Public Member Functions

- Galileo_Iono ()=default
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the iono data on disk file.

Public Attributes

- double ai0 {}

  Effective Ionisation Level 1st order parameter [sfu].
- double ai1 {}

  Effective Ionisation Level 2st order parameter [sfu/degree].
- double ai2 {}

  Effective Ionisation Level 3st order parameter [sfu/degree].
- int32_t tow {}

  UTC data reference Time of Week [s].
- int32_t WN {}

  UTC data reference Week number [week].
- bool Region1_flag {}

  Ionospheric Disturbance Flag for region 1.
- bool Region2_flag {}

  Ionospheric Disturbance Flag for region 2.
- bool Region3_flag {}

  Ionospheric Disturbance Flag for region 3.
- bool Region4_flag {}

  Ionospheric Disturbance Flag for region 4.
- bool Region5_flag {}

  Ionospheric Disturbance Flag for region 5.
- bool valid {}

  True after a complete ionospheric model has been decoded.

12.142.1 Detailed Description

This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6.
See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
Definition at line 38 of file galileo_iono.h.

12.142.2 Constructor & Destructor Documentation

12.142.2.1 Galileo_Iono()

`Galileo_Iono::Galileo_Iono () [default]`
Default constructor

12.142.3 Member Function Documentation

12.142.3.1 serialize()

```
template<class Archive>
void Galileo_Iono::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the iono data on disk file.
Definition at line 69 of file galileo_iono.h.
References ai0, ai1, ai2, Region1_flag, Region2_flag, Region3_flag, Region4_flag, Region5_flag, tow, valid, and WN.

12.142.4 Member Data Documentation

12.142.4.1 ai0

```
double Galileo_Iono::ai0 {}
```

Effective Ionisation Level 1st order parameter [sfu].
Definition at line 47 of file galileo_iono.h.
Referenced by serialize().

12.142.4.2 ai1

```
double Galileo_Iono::ai1 {}
```

Effective Ionisation Level 2st order parameter [sfu/degree].
Definition at line 48 of file galileo_iono.h.
Referenced by serialize().

12.142.4.3 ai2

```
double Galileo_Iono::ai2 {}
```

Effective Ionisation Level 3st order parameter [sfu/degree].
Definition at line 49 of file galileo_iono.h.
Referenced by serialize().

12.142.4.4 Region1_flag

```
bool Galileo_Iono::Region1_flag {}
```

Ionospheric Disturbance Flag for region 1.
Definition at line 56 of file galileo_iono.h.
Referenced by serialize().

12.142.4.5 Region2_flag

```
bool Galileo_Iono::Region2_flag {}
```

Ionospheric Disturbance Flag for region 2.
Definition at line 57 of file galileo_iono.h.
Referenced by serialize().

12.142.4.6 Region3_flag

```
bool Galileo_Iono::Region3_flag {}
```

Ionospheric Disturbance Flag for region 3.
Definition at line 58 of file galileo_iono.h.
Referenced by serialize().

12.142.4.7 Region4_flag

```
bool Galileo_Iono::Region4_flag {}
```
Ionospheric Disturbance Flag for region 4.
Definition at line 59 of file galileo_iono.h.
Referenced by serialize().

12.142.4.8 Region5_flag

```
bool Galileo_Iono::Region5_flag {}
```
Ionospheric Disturbance Flag for region 5.
Definition at line 60 of file galileo_iono.h.
Referenced by serialize().

12.142.4.9 tow

```
int32_t Galileo_Iono::tow {}
```
UTC data reference Time of Week [s].
Definition at line 52 of file galileo_iono.h.
Referenced by serialize().

12.142.4.10 valid

```
bool Galileo_Iono::valid {}
```
True after a complete ionospheric model has been decoded.
Definition at line 61 of file galileo_iono.h.
Referenced by serialize().

12.142.4.11 WN

```
int32_t Galileo_Iono::WN {}
```
UTC data reference Week number [week].
Definition at line 53 of file galileo_iono.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- galileo_iono.h

12.143 Galileo_ISM Class Reference

This class is a storage for the GALILEO Integrity Support Message as described in Galileo ICD paragraph 5.2.
`#include <galileo_ism.h>`

Public Member Functions

- Galileo_ISM ()=default
- void **set_ism_constellation_id** (uint8_t const_id)
- void **set_ism_service_level_id** (uint8_t sl_id)
- void **set_ism_wn** (uint16_t wn_ism)
- void **set_ism_t0** (uint16_t t0)
- void **set_ism_mask_msb** (bool mask_msb)
- void **set_ism_mask** (uint32_t mask)
- void **set_ism_pconst** (uint8_t pconst)
- void **set_ism_psat** (uint8_t psat)
- void **set_ism_ura** (uint8_t ura)
- void **set_ism_ure** (uint8_t ure)
- void **set_ism_bnom** (uint8_t bnom)
- void **set_ism_Tvalidity** (uint8_t tvalidity)

- bool **check_ism_crc** (const std::bitset< GALILEO_DATA_JK_BITS > &bits)
- double **get_pconst_value** () const
- double **get_psat_value** () const
- float **get_ura_m** () const
- float **get_ure_m** () const
- float **get_bnom_m** () const
- uint32_t **get_mask_ISM** () const
- uint16_t **get_WN_ISM** () const
- uint32_t **get_t0_ISM** () const
- uint16_t **get_Tvalidity_hours** () const
- bool **get_ism_mask_msb** () const
- bool **ism_parameters_apply** (uint32_t svid) const

12.143.1 Detailed Description

This class is a storage for the GALILEO Integrity Support Message as described in Galileo ICD paragraph 5.2.
See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_↩OS_SIS_ICD_v2.1.pdf
Definition at line 40 of file galileo_ism.h.

12.143.2 Constructor & Destructor Documentation

12.143.2.1 Galileo_ISM()

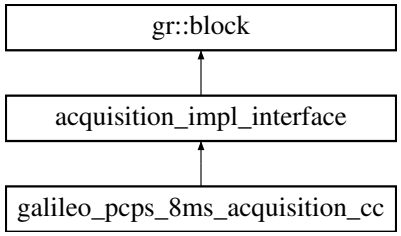
```
Galileo_ISM::Galileo_ISM ()  [default]
```

Default constructor
The documentation for this class was generated from the following file:

- galileo_ism.h

12.144 galileo_pcps_8ms_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).
`#include <galileo_pcps_8ms_acquisition_cc.h>`
Inheritance diagram for galileo_pcps_8ms_acquisition_cc:



Public Member Functions

- ~galileo_pcps_8ms_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code) override

Sets local code for PCPS acquisition algorithm.

- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.

- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.

- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.

- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- galileo_pcps_8ms_acquisition_cc_sptr **galileo_pcps_8ms_make_acquisition_cc** (const Acq_Conf &conf)

12.144.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).
Definition at line 51 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.2 Constructor & Destructor Documentation

12.144.2.1 ~galileo_pcps_8ms_acquisition_cc()

```
galileo_pcps_8ms_acquisition_cc::~galileo_pcps_8ms_acquisition_cc ()
```

Default destructor.

12.144.3 Member Function Documentation

12.144.3.1 general_work()

```
int galileo_pcps_8ms_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.144.3.2 mag()

```
uint32_t galileo_pcps_8ms_acquisition_cc::mag () const  [inline], [override], [virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 72 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.3.3 set_active()

```
void galileo_pcps_8ms_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 88 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.3.4 set_channel()

```
void galileo_pcps_8ms_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 102 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.3.5 set_channel_fsm()

```
void galileo_pcps_8ms_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 110 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.3.6 set_gnss_synchro()

```
void galileo_pcps_8ms_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 64 of file galileo_pcps_8ms_acquisition_cc.h.

12.144.3.7 set_local_code()

```
void galileo_pcps_8ms_acquisition_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- galileo_pcps_8ms_acquisition_cc.h

12.145 Galileo_Reduced_CED Class Reference

This class manages the Galileo Reduced Clock and Ephemeris Data.
`#include <galileo_reduced_ced.h>`

Public Member Functions

- Galileo_Reduced_CED ()=default
- Galileo_Ephemeris compute_eph () const
- bool is_valid_at (uint32_t week, uint32_t tow) const

  Return true when this Reduced CED set is applicable at the given GST epoch.
- uint32_t reference_time () const

  Return the Reduced CED orbit and clock reference time of week.

Public Attributes

- uint32_t PRN {}

  Satellite ID.
- uint32_t WN {}

  GST week containing the start of transmission.
- uint32_t TOTRedCED {}

  Start time of transmission of the Reduced CED word as GST TOW [s].
- double DeltaAred {}

  Difference between the Reduced CED semi-major axis and the nominal semi-major axis [meters].
- double exred {}

  Reduced CED eccentricity vector component x.
- double eyred {}

  Reduced CED eccentricity vector component y.
- double Deltai0red {}

  Difference between the Reduced CED inclination angle at reference time and the nominal inclination [rad].
- double Omega0red {}

  Reduced CED longitude of ascending node at weekly epoch [rad].
- double lambda0red {}

  Reduced CED mean argument of latitude [rad].
- double af0red {}

  Reduced CED satellite clock bias correction coefficient [seconds].
- double af1red {}

  Reduced CED satellite clock drift correction coefficient [seconds/seconds].
- double BGD_E1E5b {}

  E1-E5b Broadcast Group Delay [s].
- int32_t E5b_HS {}

  E5b Signal Health Status.
- int32_t E1B_HS {}

  E1B Signal Health Status.
- bool E5b_DVS {}

  E5b Data Validity Status.
- bool E1B_DVS {}

  E1B Data Validity Status.

Static Public Attributes

- static constexpr uint32_t CED_VALIDITY_SECONDS = 600U

12.145.1 Detailed Description

This class manages the Galileo Reduced Clock and Ephemeris Data.
Definition at line 33 of file galileo_reduced_ced.h.

12.145.2 Constructor & Destructor Documentation

12.145.2.1 Galileo_Reduced_CED()

```
Galileo_Reduced_CED::Galileo_Reduced_CED () [default]
```

Default constructor

12.145.3 Member Function Documentation

12.145.3.1 compute_eph()

```
Galileo_Ephemeris Galileo_Reduced_CED::compute_eph () const
```

Convert to Galileo_Ephemeris

12.145.3.2 is_valid_at()

```
bool Galileo_Reduced_CED::is_valid_at (
            uint32_t week,
            uint32_t tow) const
```

Return true when this Reduced CED set is applicable at the given GST epoch.

12.145.3.3 reference_time()

```
uint32_t Galileo_Reduced_CED::reference_time () const
```

Return the Reduced CED orbit and clock reference time of week.

12.145.4 Member Data Documentation

12.145.4.1 af0red

```
double Galileo_Reduced_CED::af0red {}
```

Reduced CED satellite clock bias correction coefficient [seconds].
Definition at line 67 of file galileo_reduced_ced.h.

12.145.4.2 af1red

```
double Galileo_Reduced_CED::af1red {}
```

Reduced CED satellite clock drift correction coefficient [seconds/seconds].
Definition at line 68 of file galileo_reduced_ced.h.

12.145.4.3 BGD_E1E5b

```
double Galileo_Reduced_CED::BGD_E1E5b {}
```

E1-E5b Broadcast Group Delay [s].
Definition at line 72 of file galileo_reduced_ced.h.

12.145.4.4 CED_VALIDITY_SECONDS

```
uint32_t Galileo_Reduced_CED::CED_VALIDITY_SECONDS = 600U [static], [constexpr]
```

Definition at line 56 of file galileo_reduced_ced.h.

12.145.4.5 DeltaAred

```
double Galileo_Reduced_CED::DeltaAred {}
```

Difference between the Reduced CED semi-major axis and the nominal semi-major axis [meters].
Definition at line 61 of file galileo_reduced_ced.h.

12.145.4.6 Deltai0red

```
double Galileo_Reduced_CED::Deltai0red {}
```
Difference between the Reduced CED inclination angle at reference time and the nominal inclination [rad].
Definition at line 64 of file galileo_reduced_ced.h.

12.145.4.7 E1B_DVS

```
bool Galileo_Reduced_CED::E1B_DVS {}
```
E1B Data Validity Status.
Definition at line 76 of file galileo_reduced_ced.h.

12.145.4.8 E1B_HS

```
int32_t Galileo_Reduced_CED::E1B_HS {}
```
E1B Signal Health Status.
Definition at line 74 of file galileo_reduced_ced.h.

12.145.4.9 E5b_DVS

```
bool Galileo_Reduced_CED::E5b_DVS {}
```
E5b Data Validity Status.
Definition at line 75 of file galileo_reduced_ced.h.

12.145.4.10 E5b_HS

```
int32_t Galileo_Reduced_CED::E5b_HS {}
```
E5b Signal Health Status.
Definition at line 73 of file galileo_reduced_ced.h.

12.145.4.11 exred

```
double Galileo_Reduced_CED::exred {}
```
Reduced CED eccentricity vector component x.
Definition at line 62 of file galileo_reduced_ced.h.

12.145.4.12 eyred

```
double Galileo_Reduced_CED::eyred {}
```
Reduced CED eccentricity vector component y.
Definition at line 63 of file galileo_reduced_ced.h.

12.145.4.13 lambda0red

```
double Galileo_Reduced_CED::lambda0red {}
```
Reduced CED mean argument of latitude [rad].
Definition at line 66 of file galileo_reduced_ced.h.

12.145.4.14 Omega0red

```
double Galileo_Reduced_CED::Omega0red {}
```
Reduced CED longitude of ascending node at weekly epoch [rad].
Definition at line 65 of file galileo_reduced_ced.h.

12.145.4.15 PRN

```
uint32_t Galileo_Reduced_CED::PRN {}
```
Satellite ID.
Definition at line 58 of file galileo_reduced_ced.h.

12.145.4.16 TOTRedCED

`uint32_t Galileo_Reduced_CED::TOTRedCED {}`
Start time of transmission of the Reduced CED word as GST TOW [s].
Definition at line 60 of file galileo_reduced_ced.h.

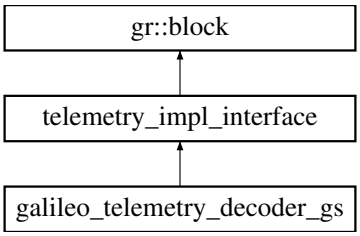
12.145.4.17 WN

`uint32_t Galileo_Reduced_CED::WN {}`
GST week containing the start of transmission.
Definition at line 59 of file galileo_reduced_ced.h.
The documentation for this class was generated from the following file:

- galileo_reduced_ced.h

12.146 galileo_telemetry_decoder_gs Class Reference

This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.
`#include <galileo_telemetry_decoder_gs.h>`
Inheritance diagram for galileo_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int32_t channel) override

  Set receiver's channel.
- void reset () override
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_channel (int channel)=0

Friends

- galileo_telemetry_decoder_gs_sptr **galileo_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int frame_type)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()

- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.146.1 Detailed Description

This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.
Definition at line 49 of file galileo_telemetry_decoder_gs.h.

12.146.2 Member Function Documentation

12.146.2.1 general_work()

```
int galileo_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.146.2.2 reset()

```
void galileo_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.146.2.3 set_channel()

```
void galileo_telemetry_decoder_gs::set_channel (
            int32_t channel)  [override]
```

Set receiver's channel.

12.146.2.4 set_satellite()

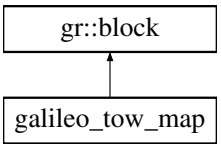
```
void galileo_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- galileo_telemetry_decoder_gs.h

12.147 galileo_tow_map Class Reference

Inheritance diagram for galileo_tow_map:



Public Member Functions

- ~galileo_tow_map ()=default

  Default destructor.

Friends

- galileo_tow_map_sptr **galileo_tow_map_make** ()

12.147.1 Detailed Description

Definition at line 62 of file galileo_tow_map.h.

12.147.2 Constructor & Destructor Documentation

12.147.2.1 ~galileo_tow_map()

```
galileo_tow_map::~galileo_tow_map () [default]
```

Default destructor.

The documentation for this class was generated from the following file:

- galileo_tow_map.h

12.148 Galileo_Utc_Model Class Reference

This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf paragraph 5.1.7.

```
#include <galileo_utc_model.h>
```

Public Member Functions

- Galileo_Utc_Model ()=default
- double GST_to_UTC_time (double t_e, int32_t WN) const

  GST-UTC Conversion Algorithm and Parameters.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the UTC data on disk file.

Static Public Member Functions

- static int32_t truncated_week_diff (int32_t wn, int32_t wn_ref, int32_t modulus)

  Signed difference between two truncated week numbers, wrapped to [-modulus/2, modulus/2). Implements the week roll-over handling required by OS SIS ICD 5.1.7 (modulus 256) and 5.1.8 (modulus 64).

Public Attributes

- double A0 {}
- double A1 {}
- int32_t Delta_tLS {}
- int32_t tot {}

  UTC data reference Time of Week [s].
- int32_t WNot {}

  UTC data reference Week number [week].
- int32_t WN_LSF {}
- int32_t DN {}
- int32_t Delta_tLSF {}
- double A_0G {}
- double A_1G {}
- int32_t t_0G {}
- int32_t WN_0G {}
- bool flag_utc_model {}
- bool flag_GGTO {}

12.148.1 Detailed Description

This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD https://www.↩ gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.↩ pdf paragraph 5.1.7.
Definition at line 37 of file galileo_utc_model.h.

12.148.2 Constructor & Destructor Documentation

12.148.2.1 Galileo_Utc_Model()

```
Galileo_Utc_Model::Galileo_Utc_Model () [default]
```

Default constructor

12.148.3 Member Function Documentation

12.148.3.1 GST_to_UTC_time()

```
double Galileo_Utc_Model::GST_to_UTC_time (
            double t_e,
            int32_t WN) const
```

GST-UTC Conversion Algorithm and Parameters.

12.148.3.2 serialize()

```
template<class Archive>
void Galileo_Utc_Model::serialize (
            Archive & archive,
            const unsigned int version) [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the UTC data on disk file.
Definition at line 80 of file galileo_utc_model.h.
References tot, and WNot.

12.148.3.3 truncated_week_diff()

```
int32_t Galileo_Utc_Model::truncated_week_diff (
            int32_t wn,
            int32_t wn_ref,
            int32_t modulus) [static]
```

Signed difference between two truncated week numbers, wrapped to [-modulus/2, modulus/2). Implements the week roll-over handling required by OS SIS ICD 5.1.7 (modulus 256) and 5.1.8 (modulus 64).

12.148.4 Member Data Documentation

12.148.4.1 A0

```
double Galileo_Utc_Model::A0 {}
```

Definition at line 56 of file galileo_utc_model.h.

12.148.4.2 A1

```
double Galileo_Utc_Model::A1 {}
```

Definition at line 57 of file galileo_utc_model.h.

12.148.4.3 A_0G

```
double Galileo_Utc_Model::A_0G {}
```

Definition at line 66 of file galileo_utc_model.h.

12.148.4.4 A_1G

```
double Galileo_Utc_Model::A_1G {}
```
Definition at line 67 of file galileo_utc_model.h.

12.148.4.5 Delta_tLS

```
int32_t Galileo_Utc_Model::Delta_tLS {}
```
Definition at line 58 of file galileo_utc_model.h.

12.148.4.6 Delta_tLSF

```
int32_t Galileo_Utc_Model::Delta_tLSF {}
```
Definition at line 63 of file galileo_utc_model.h.

12.148.4.7 DN

```
int32_t Galileo_Utc_Model::DN {}
```
Definition at line 62 of file galileo_utc_model.h.

12.148.4.8 flag_GGTO

```
bool Galileo_Utc_Model::flag_GGTO {}
```
Definition at line 72 of file galileo_utc_model.h.

12.148.4.9 flag_utc_model

```
bool Galileo_Utc_Model::flag_utc_model {}
```
Definition at line 71 of file galileo_utc_model.h.

12.148.4.10 t_0G

```
int32_t Galileo_Utc_Model::t_0G {}
```
Definition at line 68 of file galileo_utc_model.h.

12.148.4.11 tot

```
int32_t Galileo_Utc_Model::tot {}
```
UTC data reference Time of Week [s].
Definition at line 59 of file galileo_utc_model.h.
Referenced by serialize().

12.148.4.12 WN_0G

```
int32_t Galileo_Utc_Model::WN_0G {}
```
Definition at line 69 of file galileo_utc_model.h.

12.148.4.13 WN_LSF

```
int32_t Galileo_Utc_Model::WN_LSF {}
```
Definition at line 61 of file galileo_utc_model.h.

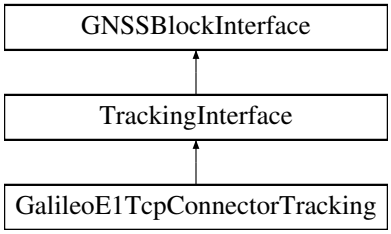
12.148.4.14 WNot

```
int32_t Galileo_Utc_Model::WNot {}
```
UTC data reference Week number [week].
Definition at line 60 of file galileo_utc_model.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- galileo_utc_model.h

12.149 GalileoE1TcpConnectorTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.
`#include <galileo_e1_tcp_connector_tracking.h>`
Inheritance diagram for GalileoE1TcpConnectorTracking:



Public Member Functions

- **GalileoE1TcpConnectorTracking** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Galileo_E1_TCP_CONNECTOR_Tracking".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override
- void stop_tracking () override

  Stop running tracking.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.149.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.
Definition at line 43 of file galileo_e1_tcp_connector_tracking.h.

12.149.2 Member Function Documentation

12.149.2.1 connect()

```
void GalileoE1TcpConnectorTracking::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.149.2.2 disconnect()

```
void GalileoE1TcpConnectorTracking::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.149.2.3 get_left_block()

```
gr::basic_block_sptr GalileoE1TcpConnectorTracking::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.149.2.4 get_right_block()

```
gr::basic_block_sptr GalileoE1TcpConnectorTracking::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.149.2.5 implementation()

```
std::string GalileoE1TcpConnectorTracking::implementation ()  [inline], [override], [virtual]
```

Returns "Galileo_E1_TCP_CONNECTOR_Tracking".
Implements GNSSBlockInterface.
Definition at line 60 of file galileo_e1_tcp_connector_tracking.h.

12.149.2.6 item_size()

```
size_t GalileoE1TcpConnectorTracking::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 65 of file galileo_e1_tcp_connector_tracking.h.

12.149.2.7 role()

```
std::string GalileoE1TcpConnectorTracking::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 54 of file galileo_e1_tcp_connector_tracking.h.

12.149.2.8 set_channel()

```
void GalileoE1TcpConnectorTracking::set_channel (
            unsigned int channel)  [override], [virtual]
```

Set tracking channel unique ID.
Implements TrackingInterface.

12.149.2.9 set_gnss_synchro()

```
void GalileoE1TcpConnectorTracking::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.149.2.10 start_tracking()

```
void GalileoE1TcpConnectorTracking::start_tracking ()  [override], [virtual]
```

Implements TrackingInterface.

12.149.2.11 stop_tracking()

```
void GalileoE1TcpConnectorTracking::stop_tracking ()  [override], [virtual]
```

Stop running tracking.
Implements TrackingInterface.

The documentation for this class was generated from the following file:

- galileo_e1_tcp_connector_tracking.h

12.150 GalileoTowMapEntry Struct Reference

Public Member Functions

- GalileoTowMapEntry (uint32_t entry_week, uint32_t entry_tow_ms, uint64_t entry_sample_counter)

Public Attributes

- uint32_t week {GALILEO_TOW_MAP_INVALID_WEEK}
- uint32_t tow_ms {GALILEO_TOW_MAP_INVALID_TOW_MS}
- uint64_t sample_counter {GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER}

12.150.1 Detailed Description

Definition at line 44 of file galileo_tow_map.h.

12.150.2 Constructor & Destructor Documentation

12.150.2.1 GalileoTowMapEntry()

```
GalileoTowMapEntry::GalileoTowMapEntry (
            uint32_t entry_week,
            uint32_t entry_tow_ms,
            uint64_t entry_sample_counter)  [inline]
```

Definition at line 47 of file galileo_tow_map.h.

12.150.3 Member Data Documentation

12.150.3.1 sample_counter

```
uint64_t GalileoTowMapEntry::sample_counter {GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER}
```

Definition at line 56 of file galileo_tow_map.h.

12.150.3.2 tow_ms

```
uint32_t GalileoTowMapEntry::tow_ms {GALILEO_TOW_MAP_INVALID_TOW_MS}
```

Definition at line 55 of file galileo_tow_map.h.

12.150.3.3 week

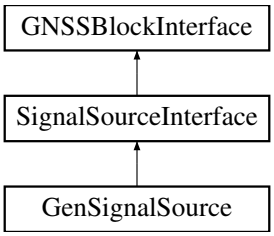
```
uint32_t GalileoTowMapEntry::week {GALILEO_TOW_MAP_INVALID_WEEK}
```

Definition at line 54 of file galileo_tow_map.h.
The documentation for this struct was generated from the following file:

- galileo_tow_map.h

12.151 GenSignalSource Class Reference

This class wraps blocks that generates synthesized GNSS signal and filters the signal.
`#include <gen_signal_source.h>`
Inheritance diagram for GenSignalSource:



Public Member Functions

- GenSignalSource (std::shared_ptr< GNSSBlockInterface > signal_generator, std::shared_ptr< GNSSBlockInterface > filter, std::string role, Concurrent_Queue< pmt::pmt_t > *queue)

  Constructor.
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string role () override
- std::string implementation () override

  Returns "Signal Source".
- size_t item_size () override
- size_t getRfChannels () const final
- std::shared_ptr< GNSSBlockInterface > signal_generator () const

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.151.1 Detailed Description

This class wraps blocks that generates synthesized GNSS signal and filters the signal.
Definition at line 41 of file gen_signal_source.h.

12.151.2 Constructor & Destructor Documentation

12.151.2.1 GenSignalSource()

```
GenSignalSource::GenSignalSource (
            std::shared_ptr< GNSSBlockInterface > signal_generator,
            std::shared_ptr< GNSSBlockInterface > filter,
            std::string role,
            Concurrent_Queue< pmt::pmt_t > * queue)
```

Constructor.

12.151.3 Member Function Documentation

12.151.3.1 connect()

```
void GenSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.151.3.2 disconnect()

```
void GenSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Implements GNSSBlockInterface.

12.151.3.3 get_left_block()

```
gr::basic_block_sptr GenSignalSource::get_left_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.151.3.4 get_right_block()

```
gr::basic_block_sptr GenSignalSource::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.151.3.5 getRfChannels()

```
size_t GenSignalSource::getRfChannels () const  [inline], [final], [virtual]
```
Implements SignalSourceInterface.
Definition at line 57 of file gen_signal_source.h.

12.151.3.6 implementation()

```
std::string GenSignalSource::implementation ()  [inline], [override], [virtual]
```
Returns "Signal Source".
Implements GNSSBlockInterface.
Definition at line 55 of file gen_signal_source.h.

12.151.3.7 item_size()

```
size_t GenSignalSource::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 56 of file gen_signal_source.h.

12.151.3.8 role()

```
std::string GenSignalSource::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 53 of file gen_signal_source.h.

12.151.3.9 signal_generator()

```
std::shared_ptr< GNSSBlockInterface > GenSignalSource::signal_generator () const  [inline]
```
Definition at line 59 of file gen_signal_source.h.
The documentation for this class was generated from the following file:

- gen_signal_source.h

12.152 Geohash Class Reference

Class for geohash encoding / decoding See https://en.wikipedia.org/wiki/Geohash.
`#include <geohash.h>`

Public Member Functions

- std::string encode (double lat, double lon, int precision=-1) const
- std::array< double, 2 > decode (std::string geohash) const

12.152.1 Detailed Description

Class for geohash encoding / decoding See https://en.wikipedia.org/wiki/Geohash.
Definition at line 34 of file geohash.h.

12.152.2 Member Function Documentation

12.152.2.1 decode()

```
std::array< double, 2 > Geohash::decode (
            std::string geohash) const
```

Decode geohash to latitude/longitude (location is approximate centre of geohash cell, to reasonable precision).

Parameters

{string}	geohash - Geohash string to be converted to latitude/longitude.

Returns

{lat, lon} (Center of) geohashed location.

Exceptions

Invalid	geohash.

12.152.2.2 encode()

```
std::string Geohash::encode (
            double lat,
            double lon,
            int precision = -1) const
```

Encodes latitude/longitude to geohash, either to specified precision or to automatically evaluated precision.

Parameters

{double}	lat - Latitude in degrees.
{double}	lon - Longitude in degrees.
{int}	[precision] - Number of characters in resulting geohash.

Returns

{string} Geohash of supplied latitude/longitude.

Exceptions

Invalid	geohash.

The documentation for this class was generated from the following file:

- geohash.h

12.153 GeoJSON_Printer Class Reference

Prints PVT solutions in GeoJSON format file.
`#include <geojson_printer.h>`

Public Member Functions

- **GeoJSON_Printer** (const std::string &base_path=".")
- bool **set_headers** (const std::string &filename, bool time_tag_name=true)
- bool **print_position** (const Pvt_Solution *const position)
- bool **close_file** ()

12.153.1 Detailed Description

Prints PVT solutions in GeoJSON format file.
See https://tools.ietf.org/html/rfc7946
Definition at line 39 of file geojson_printer.h.
The documentation for this class was generated from the following file:

- geojson_printer.h

12.154 geph_t Struct Reference

Public Attributes

- int sat
- int iode
- int frq
- int svh
- int sva
- int age
- gtime_t toe
- gtime_t tof
- double pos [3]
- double vel [3]
- double acc [3]
- double taun
- double gamn
- double dtaun

12.154.1 Detailed Description

Definition at line 498 of file rtklib.h.

12.154.2 Member Data Documentation

12.154.2.1 acc

```
double geph_t::acc[3]
```
Definition at line 508 of file rtklib.h.

12.154.2.2 age

```
int geph_t::age
```
Definition at line 503 of file rtklib.h.

12.154.2.3 dtaun

```
double geph_t::dtaun
```
Definition at line 510 of file rtklib.h.

12.154.2.4 frq

`int geph_t::frq`
Definition at line 502 of file rtklib.h.

12.154.2.5 gamn

`double geph_t::gamn`
Definition at line 509 of file rtklib.h.

12.154.2.6 iode

`int geph_t::iode`
Definition at line 501 of file rtklib.h.

12.154.2.7 pos

`double geph_t::pos[3]`
Definition at line 506 of file rtklib.h.

12.154.2.8 sat

`int geph_t::sat`
Definition at line 500 of file rtklib.h.

12.154.2.9 sva

`int geph_t::sva`
Definition at line 503 of file rtklib.h.

12.154.2.10 svh

`int geph_t::svh`
Definition at line 503 of file rtklib.h.

12.154.2.11 taun

`double geph_t::taun`
Definition at line 509 of file rtklib.h.

12.154.2.12 toe

`gtime_t geph_t::toe`
Definition at line 504 of file rtklib.h.

12.154.2.13 tof

`gtime_t geph_t::tof`
Definition at line 505 of file rtklib.h.

12.154.2.14 vel

`double geph_t::vel[3]`
Definition at line 507 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.155 Glonass_Gnav_Almanac Class Reference

This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1).
`#include <glonass_gnav_almanac.h>`

Public Member Functions

- Glonass_Gnav_Almanac ()=default
- bool satellite_position (double elapsed_s, std::array< double, 3 > &position_m) const

  Compute ECEF position in metres, elapsed_s after the reference ascending node (ICD A.3.2).
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the almanac data on disk file.

Public Attributes

- int32_t d_N_A {}

  Almanac reference day within its four-year interval (zero if unknown).
- int32_t d_N_4 {}

  Four-year interval containing d_N_A, numbered from 1996.
- double d_n_A {}

  Conventional number of satellite within GLONASS space segment [dimensionless].
- double d_H_n_A {}

  Carrier frequency number of navigation RF signal transmitted by d_nA satellite as table 4.10 (0-31) [dimensionless].
- double d_lambda_n_A {}

  Longitude of the first (within the d_NA day) ascending node of d_nA [radians].
- double d_t_lambda_n_A {}

  Time of first ascending node passage [s].
- double d_Delta_i_n_A {}

  Correction of the mean value of inclination of d_n_A satellite at instant t_lambda_n_A [radians].
- double d_Delta_T_n_A {}

  Correction to the mean value of Draconian period of d_n_A satellite at instant t_lambda_n_A [s / orbital period].
- double d_Delta_T_n_A_dot {}

  Rate of change of Draconian period of d_n_A satellite at instant t_lambda_n_A [s / orbital period^ 2].
- double d_epsilon_n_A {}

  Eccentricity of d_n_A satellite at instant t_lambda_n_A [dimensionless].
- double d_omega_n_A {}

  Argument of perigee of d_n_A satellite at instant t_lambdan_A [radians].
- double d_M_n_A {}

  Type of satellite n_A [dimensionless].
- double d_KP {}

  Notification on forthcoming leap second correction of UTC [dimensionless].
- double d_tau_n_A {}

  Coarse value of d_n_A satellite time correction to GLONASS time at instant t_lambdan_A[s].
- bool d_C_n {}

  Generalized "unhealthy flag" of n_A satellite at instant of almanac upload [dimensionless].
- bool d_l_n {}

  Health flag for nth satellite; ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
- int32_t i_satellite_freq_channel {}

  SV Frequency Channel Number.
- uint32_t PRN {}

SV PRN Number, equivalent to slot number for compatibility with GPS.

- uint32_t i_satellite_slot_number {}

  SV Slot Number.

12.155.1 Detailed Description

This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1).

Note

Code added as part of GSoC 2017 program

See also

GLONASS ICD

Definition at line 39 of file glonass_gnav_almanac.h.

12.155.2 Constructor & Destructor Documentation

12.155.2.1 Glonass_Gnav_Almanac()

```
Glonass_Gnav_Almanac::Glonass_Gnav_Almanac ()  [default]
```

Default constructor

12.155.3 Member Function Documentation

12.155.3.1 satellite_position()

```
bool Glonass_Gnav_Almanac::satellite_position (
            double elapsed_s,
            std::array< double, 3 > & position_m) const
```

Compute ECEF position in metres, elapsed_s after the reference ascending node (ICD A.3.2).

12.155.3.2 serialize()

```
template<class Archive>
void Glonass_Gnav_Almanac::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the almanac data on disk file.

Definition at line 78 of file glonass_gnav_almanac.h.

References d_C_n, d_Delta_i_n_A, d_Delta_T_n_A, d_Delta_T_n_A_dot, d_epsilon_n_A, d_H_n_A, d_KP, d_l_n, d_lambda_n_A, d_M_n_A, d_N_4, d_N_A, d_n_A, d_omega_n_A, d_t_lambda_n_A, d_tau_n_A, i_satellite_freq_channel, i_satellite_slot_number, and PRN.

12.155.4 Member Data Documentation

12.155.4.1 d_C_n

```
bool Glonass_Gnav_Almanac::d_C_n {}
```

Generalized “unhealthy flag” of n_A satellite at instant of almanac upload [dimensionless].

Definition at line 65 of file glonass_gnav_almanac.h.

Referenced by serialize().

12.155.4.2 d_Delta_i_n_A

```
double Glonass_Gnav_Almanac::d_Delta_i_n_A {}
```

Correction of the mean value of inclination of d_n_A satellite at instant t_lambda_n_A [radians].

Definition at line 57 of file glonass_gnav_almanac.h.

Referenced by serialize().

12.155.4.3 d_Delta_T_n_A

```
double Glonass_Gnav_Almanac::d_Delta_T_n_A {}
```
Correction to the mean value of Draconian period of d_n_A satellite at instant t_lambda_n_A [s / orbital period].
Definition at line 58 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.4 d_Delta_T_n_A_dot

```
double Glonass_Gnav_Almanac::d_Delta_T_n_A_dot {}
```
Rate of change of Draconian period of d_n_A satellite at instant t_lambda_n_A [s / orbital period^2].
Definition at line 59 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.5 d_epsilon_n_A

```
double Glonass_Gnav_Almanac::d_epsilon_n_A {}
```
Eccentricity of d_n_A satellite at instant t_lambda_n_A [dimensionless].
Definition at line 60 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.6 d_H_n_A

```
double Glonass_Gnav_Almanac::d_H_n_A {}
```
Carrier frequency number of navigation RF signal transmitted by d_nA satellite as table 4.10 (0-31) [dimensionless].
Definition at line 54 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.7 d_KP

```
double Glonass_Gnav_Almanac::d_KP {}
```
Notification on forthcoming leap second correction of UTC [dimensionless].
Definition at line 63 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.8 d_l_n

```
bool Glonass_Gnav_Almanac::d_l_n {}
```
Health flag for nth satellite; ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
Definition at line 66 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.9 d_lambda_n_A

```
double Glonass_Gnav_Almanac::d_lambda_n_A {}
```
Longitude of the first (within the d_NA day) ascending node of d_nA [radians].
Definition at line 55 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.10 d_M_n_A

```
double Glonass_Gnav_Almanac::d_M_n_A {}
```
Type of satellite n_A [dimensionless].
Definition at line 62 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.11 d_N_4

```
int32_t Glonass_Gnav_Almanac::d_N_4 {}
```
Four-year interval containing d_N_A, numbered from 1996.
Definition at line 48 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.12 d_N_A

```
int32_t Glonass_Gnav_Almanac::d_N_A {}
```
Almanac reference day within its four-year interval (zero if unknown).
Definition at line 47 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.13 d_n_A

```
double Glonass_Gnav_Almanac::d_n_A {}
```
Conventional number of satellite within GLONASS space segment [dimensionless].
Definition at line 53 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.14 d_omega_n_A

```
double Glonass_Gnav_Almanac::d_omega_n_A {}
```
Argument of perigee of d_n_A satellite at instant t_lambdan_A [radians].
Definition at line 61 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.15 d_t_lambda_n_A

```
double Glonass_Gnav_Almanac::d_t_lambda_n_A {}
```
Time of first ascending node passage [s].
Definition at line 56 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.16 d_tau_n_A

```
double Glonass_Gnav_Almanac::d_tau_n_A {}
```
Coarse value of d_n_A satellite time correction to GLONASS time at instant t_lambdan_A[s].
Definition at line 64 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.17 i_satellite_freq_channel

```
int32_t Glonass_Gnav_Almanac::i_satellite_freq_channel {}
```
SV Frequency Channel Number.
Definition at line 69 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.18 i_satellite_slot_number

```
uint32_t Glonass_Gnav_Almanac::i_satellite_slot_number {}
```
SV Slot Number.
Definition at line 71 of file glonass_gnav_almanac.h.
Referenced by serialize().

12.155.4.19 PRN

```
uint32_t Glonass_Gnav_Almanac::PRN {}
```
SV PRN Number, equivalent to slot number for compatibility with GPS.

Definition at line 70 of file glonass_gnav_almanac.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- glonass_gnav_almanac.h

12.156 Glonass_Gnav_Ephemeris Class Reference

This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1).
`#include <glonass_gnav_ephemeris.h>`

Public Member Functions

- Glonass_Gnav_Ephemeris ()=default
- double sv_clock_drift (double transmitTime, double timeCorrUTC)

  Sets (d_satClkDrift)and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction.
- boost::posix_time::ptime compute_GLONASS_time (double offset_time) const

  Computes the GLONASS System Time and returns a boost::posix_time::ptime object \ param offset_time Is the start of day offset to compute the time.
- boost::posix_time::ptime glot_to_utc (double offset_time, double glot2utc_corr) const

  Converts from GLONASST to UTC.
- void glot_to_gpst (double tod_offset, double glot2utc_corr, double glot2gpst_corr, int32_t *WN, double *TOW) const

  Converts from GLONASST to GPST.
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Attributes

- double d_m {}

  String number within frame [dimensionless].
- double d_t_k {}

  GLONASS Time (UTC(SU) + 3 h) referenced to the beginning of the frame within the current day [s].
- double d_t_b {}

  Reference ephemeris relative time in GLONASS Time (UTC(SU) + 3 h). Index of a time interval within current day according to UTC(SU) + 03 hours 00 min. [s].
- double d_M {}

  Type of satellite transmitting navigation signal [dimensionless].
- double d_gamma_n {}

  Relative deviation of predicted carrier frequency value of n- satellite from nominal value at the instant tb ↩ [dimensionless].
- double d_tau_n {}

  Correction to the nth satellite time (tn) relative to GLONASS time (te),.
- double d_Xn {}

  Earth-fixed coordinate x of the satellite in PZ-90.02 coordinate system [km].
- double d_Yn {}

  Earth-fixed coordinate y of the satellite in PZ-90.02 coordinate system [km].
- double d_Zn {}

  Earth-fixed coordinate z of the satellite in PZ-90.02 coordinate system [km].
- double d_VXn {}

  Earth-fixed velocity coordinate x of the satellite in PZ-90.02 coordinate system [km/s].

- double d_VYn {}

  Earth-fixed velocity coordinate y of the satellite in PZ-90.02 coordinate system [km/s].
- double d_VZn {}

  Earth-fixed velocity coordinate z of the satellite in PZ-90.02 coordinate system [km/s].
- double d_AXn {}

  Earth-fixed acceleration coordinate x of the satellite in PZ-90.02 coordinate system [km/s^2].
- double d_AYn {}

  Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s^2].
- double d_AZn {}

  Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s^2].
- double d_B_n {}

  Health flag [dimensionless].
- double d_P {}

  Technological parameter of control segment, indication the satellite operation mode in respect of time parameters [dimensionless].
- double d_N_T {}

  Current date, calendar number of day within four-year interval starting from the 1-st of January in a leap year [days].
- double d_F_T {}

  Parameter that provides the predicted satellite user range accuracy at time tb [dimensionless].
- double d_n {}

  Index of the satellite transmitting given navigation signal. It corresponds to a slot number within GLONASS constellation.
- double d_Delta_tau_n {}

  Time difference between navigation RF signal transmitted in L2 sub- band and aviation RF signal transmitted in L1 sub-band by nth satellite. [dimensionless].
- double d_E_n {}

  Characterises "age" of a current information [days].
- double d_P_1 {}

  Flag of the immediate data updating [minutes].
- bool d_P_2 {}

  Flag of oddness ("1") or evenness ("0") of the value of (tb) [dimensionless].
- bool d_P_3 {}

  Flag indicating a number of satellites for which almanac is transmitted within given frame: "1" corresponds to 5 satellites and "0" corresponds to 4 satellites [dimensionless].
- bool d_P_4 {}

  Flag to show that ephemeris parameters are present. "1" indicates that updated ephemeris or frequency/time parameters have been uploaded by the control segment [dimensionless].
- bool d_l3rd_n {}

  Health flag for nth satellite; ln = 0 indicates the n-th satellite is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
- bool d_l5th_n {}

  Health flag for nth satellite; ln = 0 indicates the n-th satellite is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
- int32_t i_satellite_freq_channel {}

  SV Frequency Channel Number.
- uint32_t PRN {}

  SV PRN Number, equivalent to slot number for compatibility with GPS.
- uint32_t i_satellite_slot_number {}

  SV Slot Number.
- double d_yr = 1972.0

  Current year.
- double d_satClkDrift {}

GLONASS clock error.

- double d_dtr {}

  relativistic clock correction term

- double d_iode {}

  Issue of data, ephemeris (Bit 0-6 of tb).

- double d_tau_c {}

  GLONASST 2 UTC correction (todo) may be eliminated.

- double d_TOW {}

  GLONASST IN GPST seconds of week.

- int32_t d_WN {}

  GLONASST IN GPST week number of the start of frame.

- double d_tod {}

  Time of Day since ephemeris where decoded.

- double d_kp_leap_correction_s {}

  UTC leap second announced by the KP word and already in effect, but not yet in the receiver's leap second table [s].

12.156.1 Detailed Description

This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1).

Note

Code added as part of GSoC 2017 program

See also

GLONASS ICD

Definition at line 40 of file glonass_gnav_ephemeris.h.

12.156.2 Constructor & Destructor Documentation

12.156.2.1 Glonass_Gnav_Ephemeris()

```
Glonass_Gnav_Ephemeris::Glonass_Gnav_Ephemeris ()  [default]
```

Default constructor

12.156.3 Member Function Documentation

12.156.3.1 compute_GLONASS_time()

```
boost::posix_time::ptime Glonass_Gnav_Ephemeris::compute_GLONASS_time (
            double offset_time) const
```

Computes the GLONASS System Time and returns a boost::posix_time::ptime object \ param offset_time Is the start of day offset to compute the time.

12.156.3.2 glot_to_gpst()

```
void Glonass_Gnav_Ephemeris::glot_to_gpst (
            double tod_offset,
            double glot2utc_corr,
            double glot2gpst_corr,
            int32_t * WN,
            double * TOW) const
```

Converts from GLONASST to GPST.

Converts from GLONASST to GPST in time of week (TOW) and week number (WN) format

Parameters

in	*tod_offset*	Is the start of day offset
in	*glot2utc_corr*	Correction from GLONASST to UTC
in	*glot2gpst_corr*	Correction from GLONASST to GPST
out	*WN*	Week Number, not in mod(1024) format
out	*TOW*	Time of Week in seconds of week

12.156.3.3 glot_to_utc()

```
boost::posix_time::ptime Glonass_Gnav_Ephemeris::glot_to_utc (
            double offset_time,
            double glot2utc_corr) const
```

Converts from GLONASST to UTC.
The function simply adjust for the 6 hrs offset between GLONASST and UTC

Parameters

in	*offset_time*	Is the start of day offset
in	*glot2utc_corr*	Correction from GLONASST to UTC

Returns

UTC time as a boost::posix_time::ptime object

12.156.3.4 serialize()

```
template<class Archive>
void Glonass_Gnav_Ephemeris::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.
< SV PRN frequency channel number
< String number within frame [dimensionless]
< Time referenced to the beginning of the frame within the current day [hours, minutes, seconds]
< Index of a time interval within current day according to UTC(SU) + 03 hours 00 min. [minutes]
< Type of satellite transmitting navigation signal [dimensionless]
< Relative deviation of predicted carrier frequency value of n- satellite from nominal value at the instant tb ↩ [dimensionless]
< Correction to the nth satellite time (tn) relative to GLONASS time (te)
< Earth-fixed coordinate x of the satellite in PZ-90.02 coordinate system [km].
< Earth-fixed coordinate y of the satellite in PZ-90.02 coordinate system [km]
< Earth-fixed coordinate z of the satellite in PZ-90.02 coordinate system [km]
< Earth-fixed velocity coordinate x of the satellite in PZ-90.02 coordinate system [km/s]
< Earth-fixed velocity coordinate y of the satellite in PZ-90.02 coordinate system [km/s]
< Earth-fixed velocity coordinate z of the satellite in PZ-90.02 coordinate system [km/s]
< Earth-fixed acceleration coordinate x of the satellite in PZ-90.02 coordinate system [km/s^2]
< Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s^2]
< Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s^2]
< Health flag [dimensionless]
< Technological parameter of control segment, indication the satellite operation mode in respect of time parameters [dimensionless]
< Current date, calendar number of day within four-year interval starting from the 1-st of January in a leap year [days]

< Parameter that provides the predicted satellite user range accuracy at time tb [dimensionless]
< Index of the satellite transmitting given navigation signal. It corresponds to a slot number within GLONASS constellation
< Time difference between navigation RF signal transmitted in L2 sub- band and aviation RF signal transmitted in L1 sub-band by nth satellite. [dimensionless]
< Characterises "age" of a current information [days]
< Flag of the immediate data updating.
< Flag of oddness ("1") or evenness ("0") of the value of (tb) [dimensionless]
< Flag indicating a number of satellites for which almanac is transmitted within given frame: "1" corresponds to 5 satellites and "0" corresponds to 4 satellites [dimensionless]
< Flag to show that ephemeris parameters are present. "1" indicates that updated ephemeris or frequency/time parameters have been uploaded by the control segment [dimensionless]
< Health flag for nth satellite; ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite [dimensionless]
< Health flag for nth satellite; ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite [dimensionless]
< UTC leap second announced by the KP word and already in effect [s]
Definition at line 129 of file glonass_gnav_ephemeris.h.
References d_AXn, d_AYn, d_AZn, d_B_n, d_Delta_tau_n, d_E_n, d_F_T, d_gamma_n, d_kp_leap_correction_s, d_l3rd_n, d_l5th_n, d_M, d_m, d_n, d_N_T, d_P, d_P_1, d_P_2, d_P_3, d_P_4, d_t_b, d_t_k, d_tau_n, d_VXn, d_VYn, d_VZn, d_Xn, d_Yn, d_Zn, i_satellite_freq_channel, i_satellite_slot_number, and PRN.

12.156.3.5 sv_clock_drift()

```
double Glonass_Gnav_Ephemeris::sv_clock_drift (
            double transmitTime,
            double timeCorrUTC)
```

Sets (*d_satClkDrift*)and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction.

12.156.4 Member Data Documentation

12.156.4.1 d_AXn

```
double Glonass_Gnav_Ephemeris::d_AXn {}
```

Earth-fixed acceleration coordinate x of the satellite in PZ-90.02 coordinate system [km/s^2].
Definition at line 60 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.2 d_AYn

```
double Glonass_Gnav_Ephemeris::d_AYn {}
```

Earth-fixed acceleration coordinate y of the satellite in PZ-90.02 coordinate system [km/s^2].
Definition at line 61 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.3 d_AZn

```
double Glonass_Gnav_Ephemeris::d_AZn {}
```

Earth-fixed acceleration coordinate z of the satellite in PZ-90.02 coordinate system [km/s^2].
Definition at line 62 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.4 d_B_n

```
double Glonass_Gnav_Ephemeris::d_B_n {}
```

Health flag [dimensionless].
Definition at line 63 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.5 d_Delta_tau_n

```
double Glonass_Gnav_Ephemeris::d_Delta_tau_n {}
```

Time difference between navigation RF signal transmitted in L2 sub- band and aviation RF signal transmitted in L1 sub-band by nth satellite. [dimensionless].
Definition at line 68 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.6 d_dtr

```
double Glonass_Gnav_Ephemeris::d_dtr {}
```

relativistic clock correction term
Definition at line 84 of file glonass_gnav_ephemeris.h.

12.156.4.7 d_E_n

```
double Glonass_Gnav_Ephemeris::d_E_n {}
```

Characterises "age" of a current information [days].
Definition at line 69 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.8 d_F_T

```
double Glonass_Gnav_Ephemeris::d_F_T {}
```

Parameter that provides the predicted satellite user range accuracy at time tb [dimensionless].
Definition at line 66 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.9 d_gamma_n

```
double Glonass_Gnav_Ephemeris::d_gamma_n {}
```

Relative deviation of predicted carrier frequency value of n- satellite from nominal value at the instant tb ↩ [dimensionless].
Definition at line 52 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.10 d_iode

```
double Glonass_Gnav_Ephemeris::d_iode {}
```

Issue of data, ephemeris (Bit 0-6 of tb).
Definition at line 85 of file glonass_gnav_ephemeris.h.

12.156.4.11 d_kp_leap_correction_s

```
double Glonass_Gnav_Ephemeris::d_kp_leap_correction_s {}
```

UTC leap second announced by the KP word and already in effect, but not yet in the receiver's leap second table [s].
Definition at line 90 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.12 d_l3rd_n

```
bool Glonass_Gnav_Ephemeris::d_l3rd_n {}
```

Health flag for nth satellite; ln = 0 indicates the n-th satellite is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
Definition at line 74 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.13 d_l5th_n

```
bool Glonass_Gnav_Ephemeris::d_l5th_n {}
```

Health flag for nth satellite; ln = 0 indicates the n-th satellite is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless].
Definition at line 75 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.14 d_M

```
double Glonass_Gnav_Ephemeris::d_M {}
```

Type of satellite transmitting navigation signal [dimensionless].
Definition at line 51 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.15 d_m

```
double Glonass_Gnav_Ephemeris::d_m {}
```

String number within frame [dimensionless].
Definition at line 48 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.16 d_n

```
double Glonass_Gnav_Ephemeris::d_n {}
```

Index of the satellite transmitting given navigation signal. It corresponds to a slot number within GLONASS constellation.
Definition at line 67 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.17 d_N_T

```
double Glonass_Gnav_Ephemeris::d_N_T {}
```

Current date, calendar number of day within four-year interval starting from the 1-st of January in a leap year [days].
Definition at line 65 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.18 d_P

```
double Glonass_Gnav_Ephemeris::d_P {}
```

Technological parameter of control segment, indication the satellite operation mode in respect of time parameters [dimensionless].
Definition at line 64 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.19 d_P_1

```
double Glonass_Gnav_Ephemeris::d_P_1 {}
```

Flag of the immediate data updating [minutes].
Definition at line 70 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.20 d_P_2

```
bool Glonass_Gnav_Ephemeris::d_P_2 {}
```

Flag of oddness ("1") or evenness ("0") of the value of (tb) [dimensionless].
Definition at line 71 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.21 d_P_3

```
bool Glonass_Gnav_Ephemeris::d_P_3 {}
```

Flag indicating a number of satellites for which almanac is transmitted within given frame: "1" corresponds to 5 satellites and "0" corresponds to 4 satellites [dimensionless].
Definition at line 72 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.22 d_P_4

```
bool Glonass_Gnav_Ephemeris::d_P_4 {}
```

Flag to show that ephemeris parameters are present. "1" indicates that updated ephemeris or frequency/time parameters have been uploaded by the control segment [dimensionless].
Definition at line 73 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.23 d_satClkDrift

```
double Glonass_Gnav_Ephemeris::d_satClkDrift {}
```

GLONASS clock error.
Definition at line 83 of file glonass_gnav_ephemeris.h.

12.156.4.24 d_t_b

```
double Glonass_Gnav_Ephemeris::d_t_b {}
```

Reference ephemeris relative time in GLONASS Time (UTC(SU) + 3 h). Index of a time interval within current day according to UTC(SU) + 03 hours 00 min. [s].
Definition at line 50 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.25 d_t_k

```
double Glonass_Gnav_Ephemeris::d_t_k {}
```

GLONASS Time (UTC(SU) + 3 h) referenced to the beginning of the frame within the current day [s].
Definition at line 49 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.26 d_tau_c

```
double Glonass_Gnav_Ephemeris::d_tau_c {}
```

GLONASST 2 UTC correction (todo) may be eliminated.
Definition at line 86 of file glonass_gnav_ephemeris.h.

12.156.4.27 d_tau_n

```
double Glonass_Gnav_Ephemeris::d_tau_n {}
```

Correction to the nth satellite time (tn) relative to GLONASS time (te),.
Definition at line 53 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.28 d_tod

```
double Glonass_Gnav_Ephemeris::d_tod {}
```

Time of Day since ephemeris where decoded.
Definition at line 89 of file glonass_gnav_ephemeris.h.

12.156.4.29 d_TOW

```
double Glonass_Gnav_Ephemeris::d_TOW {}
```

GLONASST IN GPST seconds of week.

Definition at line 87 of file glonass_gnav_ephemeris.h.

12.156.4.30 d_VXn

```
double Glonass_Gnav_Ephemeris::d_VXn {}
```

Earth-fixed velocity coordinate x of the satellite in PZ-90.02 coordinate system [km/s].
Definition at line 57 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.31 d_VYn

```
double Glonass_Gnav_Ephemeris::d_VYn {}
```

Earth-fixed velocity coordinate y of the satellite in PZ-90.02 coordinate system [km/s].
Definition at line 58 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.32 d_VZn

```
double Glonass_Gnav_Ephemeris::d_VZn {}
```

Earth-fixed velocity coordinate z of the satellite in PZ-90.02 coordinate system [km/s].
Definition at line 59 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.33 d_WN

```
int32_t Glonass_Gnav_Ephemeris::d_WN {}
```

GLONASST IN GPST week number of the start of frame.
Definition at line 88 of file glonass_gnav_ephemeris.h.

12.156.4.34 d_Xn

```
double Glonass_Gnav_Ephemeris::d_Xn {}
```

Earth-fixed coordinate x of the satellite in PZ-90.02 coordinate system [km].
Definition at line 54 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.35 d_Yn

```
double Glonass_Gnav_Ephemeris::d_Yn {}
```

Earth-fixed coordinate y of the satellite in PZ-90.02 coordinate system [km].
Definition at line 55 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.36 d_yr

```
double Glonass_Gnav_Ephemeris::d_yr = 1972.0
```

Current year.
Definition at line 82 of file glonass_gnav_ephemeris.h.

12.156.4.37 d_Zn

```
double Glonass_Gnav_Ephemeris::d_Zn {}
```

Earth-fixed coordinate z of the satellite in PZ-90.02 coordinate system [km].
Definition at line 56 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.38 i_satellite_freq_channel

`int32_t Glonass_Gnav_Ephemeris::i_satellite_freq_channel {}`
SV Frequency Channel Number.
Definition at line 79 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.39 i_satellite_slot_number

`uint32_t Glonass_Gnav_Ephemeris::i_satellite_slot_number {}`
SV Slot Number.
Definition at line 81 of file glonass_gnav_ephemeris.h.
Referenced by serialize().

12.156.4.40 PRN

`uint32_t Glonass_Gnav_Ephemeris::PRN {}`
SV PRN Number, equivalent to slot number for compatibility with GPS.
Definition at line 80 of file glonass_gnav_ephemeris.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- glonass_gnav_ephemeris.h

12.157 Glonass_Gnav_Navigation_Message Class Reference

This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1).
`#include <glonass_gnav_navigation_message.h>`

Public Member Functions

- Glonass_Gnav_Navigation_Message ()=default
- bool CRC_test (std::bitset< GLONASS_GNAV_STRING_BITS > &bits) const

  Compute CRC for GLONASS GNAV strings.
- uint32_t get_frame_number (uint32_t satellite_slot_number)

  Computes the frame number being decoded given the satellite slot number.
- Glonass_Gnav_Ephemeris get_ephemeris () const

  Obtain a GLONASS GNAV SV Ephemeris class filled with current SV data.
- Glonass_Gnav_Utc_Model get_utc_model () const

  Obtain a GLONASS GNAV UTC model parameters class filled with current SV data.
- Glonass_Gnav_Almanac get_almanac (uint32_t satellite_slot_number) const

  Returns a Glonass_Gnav_Almanac object filled with the latest navigation data received.
- bool have_new_ephemeris ()

  Returns true if a new Glonass_Gnav_Ephemeris object has arrived.
- bool have_new_utc_model ()

  Returns true if new Glonass_Gnav_Utc_Model object has arrived.
- bool have_new_almanac ()

  Returns true if new Glonass_Gnav_Almanac object has arrived.
- int32_t string_decoder (const std::string &frame_string)

  Decodes the GLONASS GNAV string.
- bool get_flag_CRC_test () const
- void set_rf_link (int32_t rf_link)
- uint32_t get_alm_satellite_slot_number () const
- bool get_flag_update_slot_number () const
- void set_flag_update_slot_number (bool flag_slot)
- bool get_flag_TOW_new () const

- void set_flag_TOW_new (bool tow_new)
- bool is_flag_TOW_set () const
- void set_flag_TOW_set (bool tow_set)
- void set_flag_ephemeris_str_1 (bool ephemeris_str_1)
- void set_flag_ephemeris_str_2 (bool ephemeris_str_2)
- void set_flag_ephemeris_str_3 (bool ephemeris_str_3)
- void set_flag_ephemeris_str_4 (bool ephemeris_str_4)

12.157.1 Detailed Description

This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1).

Note

Code added as part of GSoC 2017 program

See also

GLONASS ICD

Definition at line 46 of file glonass_gnav_navigation_message.h.

12.157.2 Constructor & Destructor Documentation

12.157.2.1 Glonass_Gnav_Navigation_Message()

```
Glonass_Gnav_Navigation_Message::Glonass_Gnav_Navigation_Message ()  [default]
```

Default constructor

12.157.3 Member Function Documentation

12.157.3.1 CRC_test()

```
bool Glonass_Gnav_Navigation_Message::CRC_test (
            std::bitset< GLONASS_GNAV_STRING_BITS > & bits) const
```

Compute CRC for GLONASS GNAV strings.

Parameters

bits	Bits of the string message where to compute CRC

12.157.3.2 get_alm_satellite_slot_number()

```
uint32_t Glonass_Gnav_Navigation_Message::get_alm_satellite_slot_number () const  [inline]
```

Definition at line 122 of file glonass_gnav_navigation_message.h.

12.157.3.3 get_almanac()

```
Glonass_Gnav_Almanac Glonass_Gnav_Navigation_Message::get_almanac (
            uint32_t satellite_slot_number) const
```

Returns a Glonass_Gnav_Almanac object filled with the latest navigation data received.

Parameters

satellite_slot_number	Slot number identifier for the satellite

Returns

Returns the Glonass_Gnav_Almanac object for the input slot number

12.157.3.4 get_ephemeris()

```
Glonass_Gnav_Ephemeris Glonass_Gnav_Navigation_Message::get_ephemeris () const  [inline]
```

Obtain a GLONASS GNAV SV Ephemeris class filled with current SV data.
Definition at line 70 of file glonass_gnav_navigation_message.h.

12.157.3.5 get_flag_CRC_test()

```
bool Glonass_Gnav_Navigation_Message::get_flag_CRC_test () const  [inline]
```

Definition at line 112 of file glonass_gnav_navigation_message.h.

12.157.3.6 get_flag_TOW_new()

```
bool Glonass_Gnav_Navigation_Message::get_flag_TOW_new () const  [inline]
```

Definition at line 137 of file glonass_gnav_navigation_message.h.

12.157.3.7 get_flag_update_slot_number()

```
bool Glonass_Gnav_Navigation_Message::get_flag_update_slot_number () const  [inline]
```

Definition at line 127 of file glonass_gnav_navigation_message.h.

12.157.3.8 get_frame_number()

```
uint32_t Glonass_Gnav_Navigation_Message::get_frame_number (
            uint32_t satellite_slot_number)
```

Computes the frame number being decoded given the satellite slot number.

Parameters

satellite_slot_number	[in] Satellite slot number identifier

Returns

Frame number being decoded, 0 if operation was not successful.

12.157.3.9 get_utc_model()

```
Glonass_Gnav_Utc_Model Glonass_Gnav_Navigation_Message::get_utc_model () const  [inline]
```

Obtain a GLONASS GNAV UTC model parameters class filled with current SV data.
Definition at line 78 of file glonass_gnav_navigation_message.h.

12.157.3.10 have_new_almanac()

```
bool Glonass_Gnav_Navigation_Message::have_new_almanac ()
```

Returns true if new Glonass_Gnav_Almanac object has arrived.

12.157.3.11 have_new_ephemeris()

```
bool Glonass_Gnav_Navigation_Message::have_new_ephemeris ()
```

Returns true if a new Glonass_Gnav_Ephemeris object has arrived.

12.157.3.12 have_new_utc_model()

```
bool Glonass_Gnav_Navigation_Message::have_new_utc_model ()
```

Returns true if new Glonass_Gnav_Utc_Model object has arrived.

12.157.3.13 is_flag_TOW_set()

```
bool Glonass_Gnav_Navigation_Message::is_flag_TOW_set () const  [inline]
```

Definition at line 147 of file glonass_gnav_navigation_message.h.

12.157.3.14 set_flag_ephemeris_str_1()

```
void Glonass_Gnav_Navigation_Message::set_flag_ephemeris_str_1 (
            bool ephemeris_str_1)  [inline]
```

Definition at line 157 of file glonass_gnav_navigation_message.h.

12.157.3.15 set_flag_ephemeris_str_2()

```
void Glonass_Gnav_Navigation_Message::set_flag_ephemeris_str_2 (
            bool ephemeris_str_2)  [inline]
```

Definition at line 162 of file glonass_gnav_navigation_message.h.

12.157.3.16 set_flag_ephemeris_str_3()

```
void Glonass_Gnav_Navigation_Message::set_flag_ephemeris_str_3 (
            bool ephemeris_str_3)  [inline]
```

Definition at line 167 of file glonass_gnav_navigation_message.h.

12.157.3.17 set_flag_ephemeris_str_4()

```
void Glonass_Gnav_Navigation_Message::set_flag_ephemeris_str_4 (
            bool ephemeris_str_4)  [inline]
```

Definition at line 172 of file glonass_gnav_navigation_message.h.

12.157.3.18 set_flag_TOW_new()

```
void Glonass_Gnav_Navigation_Message::set_flag_TOW_new (
            bool tow_new)  [inline]
```

Definition at line 142 of file glonass_gnav_navigation_message.h.

12.157.3.19 set_flag_TOW_set()

```
void Glonass_Gnav_Navigation_Message::set_flag_TOW_set (
            bool tow_set)  [inline]
```

Definition at line 152 of file glonass_gnav_navigation_message.h.

12.157.3.20 set_flag_update_slot_number()

```
void Glonass_Gnav_Navigation_Message::set_flag_update_slot_number (
            bool flag_slot)  [inline]
```

Definition at line 132 of file glonass_gnav_navigation_message.h.

12.157.3.21 set_rf_link()

```
void Glonass_Gnav_Navigation_Message::set_rf_link (
            int32_t rf_link)  [inline]
```

Definition at line 117 of file glonass_gnav_navigation_message.h.

12.157.3.22 string_decoder()

```
int32_t Glonass_Gnav_Navigation_Message::string_decoder (
            const std::string & frame_string)
```

Decodes the GLONASS GNAV string.

Parameters

frame_string	[in] is the string message within the parsed frame

Returns

Returns the ID of the decoded string

The documentation for this class was generated from the following file:

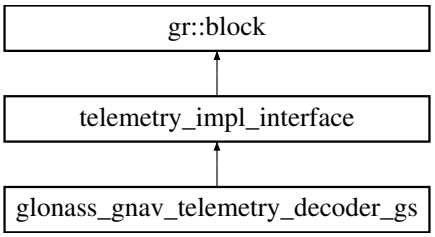
- glonass_gnav_navigation_message.h

12.158 glonass_gnav_telemetry_decoder_gs Class Reference

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals (frequency_band = 1 or 2, respectively).

`#include <glonass_gnav_telemetry_decoder_gs.h>`

Inheritance diagram for glonass_gnav_telemetry_decoder_gs:



Public Member Functions

- ~glonass_gnav_telemetry_decoder_gs () override

  Class destructor.
- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int32_t channel) override

  Set receiver's channel.
- void reset () override

  Clear synchronization state on channel re-acquisition.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_channel (int channel)=0

Friends

- glonass_gnav_telemetry_decoder_gs_sptr **glonass_gnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.158.1 Detailed Description

This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals (frequency_band = 1 or 2, respectively).

Note

Code added as part of GSoC 2017 program

See also

GLONASS ICD

Definition at line 51 of file glonass_gnav_telemetry_decoder_gs.h.

12.158.2 Constructor & Destructor Documentation

12.158.2.1 ~glonass_gnav_telemetry_decoder_gs()

```
glonass_gnav_telemetry_decoder_gs::~glonass_gnav_telemetry_decoder_gs ()  [override]
```

Class destructor.

12.158.3 Member Function Documentation

12.158.3.1 general_work()

```
int glonass_gnav_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.158.3.2 reset()

```
void glonass_gnav_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Clear synchronization state on channel re-acquisition.
Implements telemetry_impl_interface.

12.158.3.3 set_channel()

```
void glonass_gnav_telemetry_decoder_gs::set_channel (
            int32_t channel)  [override]
```

Set receiver's channel.

12.158.3.4 set_satellite()

```
void glonass_gnav_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- glonass_gnav_telemetry_decoder_gs.h

12.159 Glonass_Gnav_Utc_Model Class Reference

This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1).
`#include <glonass_gnav_utc_model.h>`

Public Member Functions

- Glonass_Gnav_Utc_Model ()=default
- double utc_time (double glonass_time_corrected) const

  Computes the Coordinated Universal Time (UTC) and returns it in [s] (GLONASS ICD (Edition 5.1) Section 3.3.3 GLONASS Time).
- int32_t announced_leap_second () const

  Returns the UTC leap second correction announced by the KP word for the end of the current quarter: +1 s, -1 s, or 0 if no correction is planned (GLONASS ICD Edition 5.1, Table 4.12).
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the almanac data on disk file.

Public Attributes

- bool valid {}
- double d_tau_c {}

  GLONASS time scale correction to UTC(SU) time. [s].
- double d_tau_gps {}

  Correction to GPS time to GLONASS time [day].
- double d_N_4 {}

  Four year interval number starting from 1996 [4 year interval].
- double d_N_A {}

  Calendar day number within the four-year period beginning since the leap year for Almanac data [days].
- double d_B1 {}

  Coefficient to determine DeltaUT1 [s].
- double d_B2 {}

  Coefficient to determine DeltaUT1 [s/msd].
- double d_KP {}

  Notification of forthcoming leap second correction of UTC (GLONASS ICD Table 4.12) [dimensionless].

12.159.1 Detailed Description

This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1).

Note

Code added as part of GSoC 2017 program

See also

GLONASS ICD

Definition at line 37 of file glonass_gnav_utc_model.h.

12.159.2 Constructor & Destructor Documentation

12.159.2.1 Glonass_Gnav_Utc_Model()

```
Glonass_Gnav_Utc_Model::Glonass_Gnav_Utc_Model ()  [default]
```

Default constructor

12.159.3 Member Function Documentation

12.159.3.1 announced_leap_second()

```
int32_t Glonass_Gnav_Utc_Model::announced_leap_second () const  [inline]
```

Returns the UTC leap second correction announced by the KP word for the end of the current quarter: +1 s, -1 s, or 0 if no correction is planned (GLONASS ICD Edition 5.1, Table 4.12).
Definition at line 66 of file glonass_gnav_utc_model.h.
References d_KP.

12.159.3.2 serialize()

```
template<class Archive>
void Glonass_Gnav_Utc_Model::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the almanac data on disk file.
Definition at line 84 of file glonass_gnav_utc_model.h.
References d_B1, d_B2, d_KP, d_N_4, d_N_A, d_tau_c, and d_tau_gps.

12.159.3.3 utc_time()

```
double Glonass_Gnav_Utc_Model::utc_time (
            double glonass_time_corrected) const
```

Computes the Coordinated Universal Time (UTC) and returns it in [s] (GLONASS ICD (Edition 5.1) Section 3.3.3 GLONASS Time).

12.159.4 Member Data Documentation

12.159.4.1 d_B1

```
double Glonass_Gnav_Utc_Model::d_B1 {}
```

Coefficient to determine DeltaUT1 [s].
Definition at line 51 of file glonass_gnav_utc_model.h.
Referenced by serialize().

12.159.4.2 d_B2

```
double Glonass_Gnav_Utc_Model::d_B2 {}
```

Coefficient to determine DeltaUT1 [s/msd].
Definition at line 52 of file glonass_gnav_utc_model.h.
Referenced by serialize().

12.159.4.3 d_KP

```
double Glonass_Gnav_Utc_Model::d_KP {}
```

Notification of forthcoming leap second correction of UTC (GLONASS ICD Table 4.12) [dimensionless].
Definition at line 53 of file glonass_gnav_utc_model.h.
Referenced by announced_leap_second(), and serialize().

12.159.4.4 d_N_4

```
double Glonass_Gnav_Utc_Model::d_N_4 {}
```

Four year interval number starting from 1996 [4 year interval].
Definition at line 49 of file glonass_gnav_utc_model.h.
Referenced by serialize().

12.159.4.5 d_N_A

```
double Glonass_Gnav_Utc_Model::d_N_A {}
```
Calendar day number within the four-year period beginning since the leap year for Almanac data [days].
Definition at line 50 of file glonass_gnav_utc_model.h.
Referenced by serialize().

12.159.4.6 d_tau_c

```
double Glonass_Gnav_Utc_Model::d_tau_c {}
```
GLONASS time scale correction to UTC(SU) time. [s].
Definition at line 47 of file glonass_gnav_utc_model.h.
Referenced by serialize().

12.159.4.7 d_tau_gps

```
double Glonass_Gnav_Utc_Model::d_tau_gps {}
```
Correction to GPS time to GLONASS time [day].
Definition at line 48 of file glonass_gnav_utc_model.h.
Referenced by serialize().

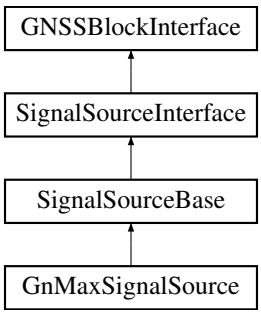
12.159.4.8 valid

```
bool Glonass_Gnav_Utc_Model::valid {}
```
Definition at line 45 of file glonass_gnav_utc_model.h.
The documentation for this class was generated from the following file:

- glonass_gnav_utc_model.h

12.160 GnMaxSignalSource Class Reference

This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.
`#include <gnmax_signal_source.h>`
Inheritance diagram for GnMaxSignalSource:



Public Member Functions

- **GnMaxSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in↩ _stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.160.1 Detailed Description

This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.
Definition at line 44 of file gnmax_signal_source.h.

12.160.2 Member Function Documentation

12.160.2.1 connect()

```
void GnMaxSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.160.2.2 disconnect()

```
void GnMaxSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.160.2.3 get_left_block()

```
gr::basic_block_sptr GnMaxSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.160.2.4 get_right_block()

```
gr::basic_block_sptr GnMaxSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.160.2.5 item_size()

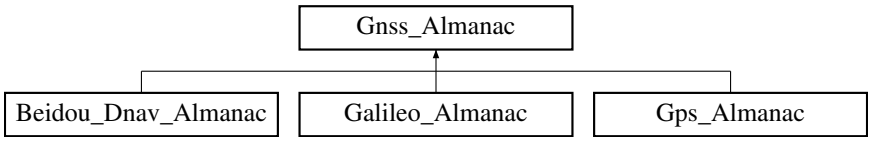
```
size_t GnMaxSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 53 of file gnmax_signal_source.h.
The documentation for this class was generated from the following file:

- gnmax_signal_source.h

12.161 Gnss_Almanac Class Reference

Base class for GNSS almanac storage.
`#include <gnss_almanac.h>`
Inheritance diagram for Gnss_Almanac:



Public Member Functions

- Gnss_Almanac ()=default
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosVelComputation (double transmitTime, std::array< double, 7 > &pos_vel_dtr) const

  Computes satellite Position and Velocity, in ECEF, for a given time (expressed in seconds of week).

Public Attributes

- uint32_t PRN {}

  SV PRN NUMBER.
- double delta_i {}

  Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles).
- int32_t toa {}

  Almanac data reference time of week [s].
- int32_t WNa {}

  Almanac week number.
- double M_0 {}

  Mean Anomaly at Reference Time [semi-circles].
- double ecc {}

  Eccentricity [dimensionless].
- double sqrtA {}

  Square Root of the Semi-Major Axis [sqrt(m)].
- double OMEGA_0 {}

  Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].
- double omega {}

  Argument of Perigee [semi-cicles].
- double OMEGAdot {}

  Rate of Right Ascension [semi-circles/s].
- double af0 {}

  Coefficient 0 of code phase offset model [s].
- double af1 {}

  Coefficient 1 of code phase offset model [s/s].
- bool from_startup_load {false}

  Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the broadcast. Runtime-only provenance flag, intentionally not serialized.

Protected Attributes

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J': QZSS.

12.161.1 Detailed Description

Base class for GNSS almanac storage.
Definition at line 33 of file gnss_almanac.h.

12.161.2 Constructor & Destructor Documentation

12.161.2.1 Gnss_Almanac()

```
Gnss_Almanac::Gnss_Almanac ()  [default]
```

Default constructor

12.161.3 Member Function Documentation

12.161.3.1 predicted_doppler()

```
double Gnss_Almanac::predicted_doppler (
            double rx_time_s,
            double lat,
            double lon,
            double h,
            double ve,
            double vn,
            double vu,
            int band) const
```

Computes prediction of the Doppler shift for a given time and receiver's position and velocity.

$$f_d = -\mathbf{v}\frac{\mathbf{x}^T}{|\mathbf{x}|}\frac{f_L}{c}$$

where:

$$\mathbf{v} = \mathbf{v}_{sat} - \mathbf{v}_{rx}$$

$$\mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{rx}$$

$$|\mathbf{x}| = \sqrt{\mathbf{x}\mathbf{x}^T}$$

Parameters

in	*rx_time_s*	Time of Week in seconds
in	*lat*	Receiver's latitude in degrees
in	*lon*	Receiver's longitude in degrees
in	*h*	Receiver's height in meters
in	*ve*	Receiver's velocity in the East direction [m/s]
in	*vn*	Receiver's velocity in the North direction [m/s]
in	*vu*	Receiver's velocity in the Up direction [m/s]
in	*band*	Signal band for which the Doppler will be computed (1: L1 C/A, E1B, BI1; 2: L2C, BI2; 3: BI3; 5: L5/E5a; 6: E6B; 7: E5b; 8: E5a+E5b)

12.161.3.2 satellitePosVelComputation()

```
void Gnss_Almanac::satellitePosVelComputation (
            double transmitTime,
            std::array< double, 7 > & pos_vel_dtr) const
```

Computes satellite Position and Velocity, in ECEF, for a given time (expressed in seconds of week).

12.161.4 Member Data Documentation

12.161.4.1 af0

```
double Gnss_Almanac::af0 {}
```

Coefficient 0 of code phase offset model [s].
Definition at line 91 of file gnss_almanac.h.

12.161.4.2 af1

```
double Gnss_Almanac::af1 {}
```

Coefficient 1 of code phase offset model [s/s].
Definition at line 92 of file gnss_almanac.h.

12.161.4.3 delta_i

```
double Gnss_Almanac::delta_i {}
```

Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles).
Definition at line 82 of file gnss_almanac.h.

12.161.4.4 ecc

```
double Gnss_Almanac::ecc {}
```

Eccentricity [dimensionless].
Definition at line 86 of file gnss_almanac.h.

12.161.4.5 from_startup_load

```
bool Gnss_Almanac::from_startup_load {false}
```

Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the broadcast. Runtime-only provenance flag, intentionally not serialized.
Definition at line 96 of file gnss_almanac.h.

12.161.4.6 M_0

```
double Gnss_Almanac::M_0 {}
```

Mean Anomaly at Reference Time [semi-circles].
Definition at line 85 of file gnss_almanac.h.

12.161.4.7 omega

```
double Gnss_Almanac::omega {}
```

Argument of Perigee [semi-cicles].
Definition at line 89 of file gnss_almanac.h.

12.161.4.8 OMEGA_0

```
double Gnss_Almanac::OMEGA_0 {}
```

Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].
Definition at line 88 of file gnss_almanac.h.

12.161.4.9 OMEGAdot

`double Gnss_Almanac::OMEGAdot {}`
Rate of Right Ascension [semi-circles/s].
Definition at line 90 of file gnss_almanac.h.

12.161.4.10 PRN

`uint32_t Gnss_Almanac::PRN {}`
SV PRN NUMBER.
Definition at line 81 of file gnss_almanac.h.

12.161.4.11 sqrtA

`double Gnss_Almanac::sqrtA {}`
Square Root of the Semi-Major Axis [sqrt(m)].
Definition at line 87 of file gnss_almanac.h.

12.161.4.12 System

`char Gnss_Almanac::System {} [protected]`
Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J': QZSS.
Definition at line 99 of file gnss_almanac.h.
Referenced by Beidou_Dnav_Almanac::Beidou_Dnav_Almanac(), Galileo_Almanac::Galileo_Almanac(), and Gps_Almanac::Gps_Almanac().

12.161.4.13 toa

`int32_t Gnss_Almanac::toa {}`
Almanac data reference time of week [s].
Definition at line 83 of file gnss_almanac.h.

12.161.4.14 WNa

`int32_t Gnss_Almanac::WNa {}`
Almanac week number.
Definition at line 84 of file gnss_almanac.h.
The documentation for this class was generated from the following file:

- gnss_almanac.h

12.162 Gnss_circular_deque< T > Class Template Reference

Public Member Functions

- Gnss_circular_deque ()
  Default constructor.
- Gnss_circular_deque (unsigned int max_size, unsigned int nchann)
  nchann = number of channels; max_size = channel capacity
- unsigned int size (unsigned int ch) const
  Returns the number of available elements in a channel.
- T & at (unsigned int ch, unsigned int pos)
  Returns a reference to an element with bound checking.
- const T & get (unsigned int ch, unsigned int pos) const
  Returns a const reference to an element without bound checking.
- T & front (unsigned int ch)
  Returns a reference to the first element in the deque.
- T & back (unsigned int ch)

Returns a reference to the last element in the deque.

- void push_back (unsigned int ch, const T &new_data)

  Inserts an element at the end of the deque.
- void pop_front (unsigned int ch)

  Removes the first element of the deque.
- void clear (unsigned int ch)

  Removes all the elements of the deque (Sets size to 0). Capacity is not modified.
- void reset (unsigned int max_size, unsigned int nchann)

  Removes all the elements in all the channels. Re-sets the number of channels and their capacity.
- void reset ()

  Removes all the channels (Sets nchann to 0).

12.162.1 Detailed Description

template<class T>
class Gnss_circular_deque< T >

Definition at line 31 of file gnss_circular_deque.h.
The documentation for this class was generated from the following file:

- gnss_circular_deque.h

12.163 Gnss_Crypto Class Reference

Class implementing cryptographic functions for Navigation Message Authentication.
`#include <gnss_crypto.h>`

Public Member Functions

- Gnss_Crypto ()

  Default constructor.
- Gnss_Crypto (const std::string &certFilePath, const std::string &merkleTreePath)
- ~Gnss_Crypto () noexcept

  Default destructor.
- bool have_public_key () const

  Returns true if the ECDSA Public Key is already loaded.
- void clear_public_key ()

  Clears the loaded ECDSA Public Key.
- bool store_public_key (const std::string &pubKeyFilePath) const
- bool verify_signature_ecdsa_p256 (const std::vector< uint8_t > &message, const std::vector< uint8_t > &signature) const

  Verify ECDSA-P256 signature (message in plain hex, signature in raw format).
- bool verify_signature_ecdsa_p521 (const std::vector< uint8_t > &message, const std::vector< uint8_t > &signature) const

  Verify ECDSA-P521 signature (message in plain hex, signature in raw format).
- std::vector< uint8_t > compute_SHA_256 (const std::vector< uint8_t > &input) const

  Computes SHA-256 hash.
- std::vector< uint8_t > compute_SHA3_256 (const std::vector< uint8_t > &input) const

  Computes SHA3-256 hash.
- std::vector< uint8_t > compute_HMAC_SHA_256 (const std::vector< uint8_t > &key, const std::vector< uint8_t > &input) const

  Computes HMAC-SHA-256 message authentication code.
- std::vector< uint8_t > compute_CMAC_AES (const std::vector< uint8_t > &key, const std::vector< uint8_t > &input) const

Computes CMAC-AES message authentication code.

- std::vector< uint8_t > get_merkle_root () const

  Gets the Merkle Tree root node ($x_{4,0}$).
- std::string get_public_key_type () const

  Gets the ECDSA Public Key type (ECDSA P-256 / ECDSA P-521 / Unknown).
- std::vector< uint8_t > get_public_key_compressed () const

  Gets the ECDSA Public Key in compressed format.
- std::string **get_merkle_tree_hash_function** () const
- void set_public_key (const std::vector< uint8_t > &publickey)

  Sets the ECDSA Public Key (publickey compressed format).
- void set_public_key_type (const std::string &public_key_type)

  Sets the ECDSA Public Key type (ECDSA P-256 / ECDSA P-521).
- void set_merkle_root (const std::vector< uint8_t > &v)

  Sets the Merkle Tree root node x($x_{4,0}$).
- void **set_merkle_tree_hash_function** (const std::string &hash_function)
- Osnma_Merkle_Tree_Material **read_merkle_xml** (const std::string &merkleFilePath)

12.163.1 Detailed Description

Class implementing cryptographic functions for Navigation Message Authentication.
Definition at line 42 of file gnss_crypto.h.

12.163.2 Constructor & Destructor Documentation

12.163.2.1 Gnss_Crypto() [1/2]

```
Gnss_Crypto::Gnss_Crypto ()
```

Default constructor.

12.163.2.2 Gnss_Crypto() [2/2]

```
Gnss_Crypto::Gnss_Crypto (
            const std::string & certFilePath,
            const std::string & merkleTreePath)
```

Constructor with a .crt or .pem file for the ECDSA Public Key and a XML file for the Merkle Tree root. Files can be downloaded by registering at https://www.gsc-europa.eu/

12.163.2.3 ~Gnss_Crypto()

```
Gnss_Crypto::~Gnss_Crypto ()  [noexcept]
```

Default destructor.

12.163.3 Member Function Documentation

12.163.3.1 clear_public_key()

```
void Gnss_Crypto::clear_public_key ()
```

Clears the loaded ECDSA Public Key.
References clear_public_key().
Referenced by clear_public_key().

12.163.3.2 compute_CMAC_AES()

```
std::vector< uint8_t > Gnss_Crypto::compute_CMAC_AES (
            const std::vector< uint8_t > & key,
            const std::vector< uint8_t > & input) const
```

Computes CMAC-AES message authentication code.
References compute_CMAC_AES().
Referenced by compute_CMAC_AES().

12.163.3.3 compute_HMAC_SHA_256()

```
std::vector< uint8_t > Gnss_Crypto::compute_HMAC_SHA_256 (
            const std::vector< uint8_t > & key,
            const std::vector< uint8_t > & input) const
```

Computes HMAC-SHA-256 message authentication code.
References compute_HMAC_SHA_256().
Referenced by compute_HMAC_SHA_256().

12.163.3.4 compute_SHA3_256()

```
std::vector< uint8_t > Gnss_Crypto::compute_SHA3_256 (
            const std::vector< uint8_t > & input) const
```

Computes SHA3-256 hash.
References compute_SHA3_256().
Referenced by compute_SHA3_256().

12.163.3.5 compute_SHA_256()

```
std::vector< uint8_t > Gnss_Crypto::compute_SHA_256 (
            const std::vector< uint8_t > & input) const
```

Computes SHA-256 hash.
References compute_SHA_256().
Referenced by compute_SHA_256().

12.163.3.6 get_merkle_root()

```
std::vector< uint8_t > Gnss_Crypto::get_merkle_root () const
```

Gets the Merkle Tree root node ($x_{4,0}$).
References get_merkle_root().
Referenced by get_merkle_root().

12.163.3.7 get_public_key_compressed()

```
std::vector< uint8_t > Gnss_Crypto::get_public_key_compressed () const
```

Gets the ECDSA Public Key in compressed format.
References get_public_key_compressed().
Referenced by get_public_key_compressed().

12.163.3.8 get_public_key_type()

```
std::string Gnss_Crypto::get_public_key_type () const
```

Gets the ECDSA Public Key type (ECDSA P-256 / ECDSA P-521 / Unknown).
References get_public_key_type().
Referenced by get_public_key_type().

12.163.3.9 have_public_key()

```
bool Gnss_Crypto::have_public_key () const
```

Returns true if the ECDSA Public Key is already loaded.
References have_public_key().
Referenced by have_public_key().

12.163.3.10 set_merkle_root()

```
void Gnss_Crypto::set_merkle_root (
            const std::vector< uint8_t > & v)
```

Sets the Merkle Tree root node x($x_{4,0}$).
References set_merkle_root().
Referenced by set_merkle_root().

12.163.3.11 set_public_key()

```
void Gnss_Crypto::set_public_key (
            const std::vector< uint8_t > & publickey)
```

Sets the ECDSA Public Key (publickey compressed format).
References set_public_key().
Referenced by set_public_key().

12.163.3.12 set_public_key_type()

```
void Gnss_Crypto::set_public_key_type (
            const std::string & public_key_type)
```

Sets the ECDSA Public Key type (ECDSA P-256 / ECDSA P-521).
References set_public_key_type().
Referenced by set_public_key_type().

12.163.3.13 store_public_key()

```
bool Gnss_Crypto::store_public_key (
            const std::string & pubKeyFilePath) const
```

Stores the ECDSA Public Key in a .pem file, which is read in a following run if the .crt file is not found
References store_public_key().
Referenced by store_public_key().

12.163.3.14 verify_signature_ecdsa_p256()

```
bool Gnss_Crypto::verify_signature_ecdsa_p256 (
            const std::vector< uint8_t > & message,
            const std::vector< uint8_t > & signature) const
```

Verify ECDSA-P256 signature (message in plain hex, signature in raw format).
References verify_signature_ecdsa_p256().
Referenced by verify_signature_ecdsa_p256().

12.163.3.15 verify_signature_ecdsa_p521()

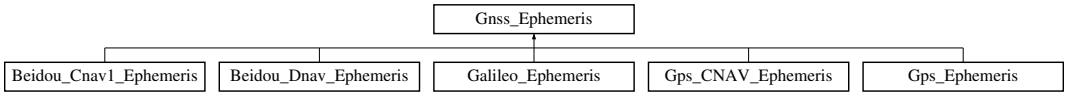
```
bool Gnss_Crypto::verify_signature_ecdsa_p521 (
            const std::vector< uint8_t > & message,
            const std::vector< uint8_t > & signature) const
```

Verify ECDSA-P521 signature (message in plain hex, signature in raw format).
References verify_signature_ecdsa_p521().
Referenced by verify_signature_ecdsa_p521().
The documentation for this class was generated from the following file:

- gnss_crypto.h

12.164 Gnss_Ephemeris Class Reference

Base class for GNSS ephemeris storage.
`#include <gnss_ephemeris.h>`
Inheritance diagram for Gnss_Ephemeris:



Public Member Functions

- double sv_clock_drift (double transmitTime)

Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).

- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

  Square root of the semi-major axis [meters^ 1/2].
- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
- double OMEGA_0 {}

  Longitude of ascending node of orbital plane at weekly epoch [rad].
- double i_0 {}

  Inclination angle at reference time [rad].
- double omega {}

  Argument of perigee [rad].
- double OMEGAdot {}

  Rate of right ascension [rad/sec].
- double idot {}

  Rate of inclination angle [rad/sec].
- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].
- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].
- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].
- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].
- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].
- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].
- int32_t toe {}

  Ephemeris reference time [s].
- int32_t toc {}

  Clock correction data reference Time of Week [sec].

- double af0 {}

  SV clock bias correction coefficient [s].
- double af1 {}

  SV clock drift correction coefficient [s/s].
- double af2 {}

  SV clock drift rate correction coefficient [s/s^ 2].
- double satClkDrift {}

  SV clock drift.
- double dtr {}

  Relativistic clock correction term.
- int32_t WN {}

  Week number.
- int32_t tow {}

  Time of Week.
- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.
- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.
- double satpos_Z {}

  Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).
- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].
- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].
- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Protected Attributes

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.164.1 Detailed Description

Base class for GNSS ephemeris storage.
Definition at line 28 of file gnss_ephemeris.h.

12.164.2 Member Function Documentation

12.164.2.1 predicted_doppler()

```
double Gnss_Ephemeris::predicted_doppler (
            double rx_time_s,
            double lat,
            double lon,
            double h,
            double ve,
            double vn,
            double vu,
            int band) const
```

Computes prediction of the Doppler shift for a given time and receiver's position and velocity.

$$f_d = -\mathbf{v}\frac{\mathbf{x}^T}{|\mathbf{x}|}\frac{f_L}{c}$$

where:

$$\mathbf{v} = \mathbf{v}_{sat} - \mathbf{v}_{rx}$$

$$\mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{rx}$$

$$|\mathbf{x}| = \sqrt{\mathbf{x}\mathbf{x}^T}$$

Parameters

in	*rx_time_s*	Time of Week in seconds
in	*lat*	Receiver's latitude in degrees
in	*lon*	Receiver's longitude in degrees
in	*h*	Receiver's height in meters
in	*ve*	Receiver's velocity in the East direction [m/s]
in	*vn*	Receiver's velocity in the North direction [m/s]
in	*vu*	Receiver's velocity in the Up direction [m/s]
in	*band*	Signal band for which the Doppler will be computed (1: L1 C/A, E1B, BI1; 2: L2C, BI2; 3: BI3; 5: L5/E5a; 6: E6B; 7: E5b; 8: E5a+E5b)

12.164.2.2 satellitePosition()

```
void Gnss_Ephemeris::satellitePosition (
            double transmitTime)
```

Computes the ECEF SV coordinates and ECEF velocity.

12.164.2.3 sv_clock_drift()

```
double Gnss_Ephemeris::sv_clock_drift (
            double transmitTime)
```

Sets (*satClkDrift*) and (*dtr*), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).

12.164.3 Member Data Documentation

12.164.3.1 A0

```
double Gnss_Ephemeris::A0 {}
```

Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
Definition at line 77 of file gnss_ephemeris.h.

12.164.3.2 Adot

```
double Gnss_Ephemeris::Adot {}
```

Change rate in semi-major axis (CNAV).
Definition at line 72 of file gnss_ephemeris.h.
Referenced by Gps_CNAV_Ephemeris::serialize().

12.164.3.3 af0

```
double Gnss_Ephemeris::af0 {}
```

SV clock bias correction coefficient [s].
Definition at line 93 of file gnss_ephemeris.h.

Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.4 af1

```
double Gnss_Ephemeris::af1 {}
```

SV clock drift correction coefficient [s/s].
Definition at line 94 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.5 af2

```
double Gnss_Ephemeris::af2 {}
```

SV clock drift rate correction coefficient [s/s^2].
Definition at line 95 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.6 Cic

```
double Gnss_Ephemeris::Cic {}
```

Amplitude of the cosine harmonic correction term to the angle of inclination [rad].
Definition at line 87 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.7 Cis

```
double Gnss_Ephemeris::Cis {}
```

Amplitude of the sine harmonic correction term to the angle of inclination [rad].
Definition at line 88 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.8 Crc

```
double Gnss_Ephemeris::Crc {}
```

Amplitude of the cosine harmonic correction term to the orbit radius [meters].
Definition at line 85 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.9 Crs

```
double Gnss_Ephemeris::Crs {}
```

Amplitude of the sine harmonic correction term to the orbit radius [meters].
Definition at line 86 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.10 Cuc

```
double Gnss_Ephemeris::Cuc {}
```

Amplitude of the cosine harmonic correction term to the argument of latitude [rad].
Definition at line 83 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.11 Cus

`double Gnss_Ephemeris::Cus {}`
Amplitude of the sine harmonic correction term to the argument of latitude [rad].
Definition at line 84 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.12 delta_n

`double Gnss_Ephemeris::delta_n {}`
Mean motion difference from computed value [rad/sec].
Definition at line 74 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.13 delta_ndot

`double Gnss_Ephemeris::delta_ndot {}`
Rate of mean motion difference from computed value (CNAV).
Definition at line 73 of file gnss_ephemeris.h.
Referenced by Gps_CNAV_Ephemeris::serialize().

12.164.3.14 dtr

`double Gnss_Ephemeris::dtr {}`
Relativistic clock correction term.
Definition at line 98 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.15 ecc

`double Gnss_Ephemeris::ecc {}`
Eccentricity.
Definition at line 75 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.16 i_0

`double Gnss_Ephemeris::i_0 {}`
Inclination angle at reference time [rad].
Definition at line 79 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.17 idot

`double Gnss_Ephemeris::idot {}`
Rate of inclination angle [rad/sec].
Definition at line 82 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.18 M_0

`double Gnss_Ephemeris::M_0 {}`
Mean anomaly at reference time [rad].
Definition at line 71 of file gnss_ephemeris.h.

Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.19 omega

```
double Gnss_Ephemeris::omega {}
```

Argument of perigee [rad].
Definition at line 80 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.20 OMEGA_0

```
double Gnss_Ephemeris::OMEGA_0 {}
```

Longitude of ascending node of orbital plane at weekly epoch [rad].
Definition at line 78 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.21 OMEGAdot

```
double Gnss_Ephemeris::OMEGAdot {}
```

Rate of right ascension [rad/sec].
Definition at line 81 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.22 PRN

```
uint32_t Gnss_Ephemeris::PRN {}
```

SV ID.
Definition at line 70 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.23 satClkDrift

```
double Gnss_Ephemeris::satClkDrift {}
```

SV clock drift.
Definition at line 97 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.24 satpos_X

```
double Gnss_Ephemeris::satpos_X {}
```

Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.
Definition at line 105 of file gnss_ephemeris.h.

12.164.3.25 satpos_Y

```
double Gnss_Ephemeris::satpos_Y {}
```

Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.
Definition at line 106 of file gnss_ephemeris.h.

12.164.3.26 satpos_Z

`double Gnss_Ephemeris::satpos_Z {}`
Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).
Definition at line 107 of file gnss_ephemeris.h.

12.164.3.27 satvel_X

`double Gnss_Ephemeris::satvel_X {}`
Earth-fixed velocity coordinate x of the satellite [m].
Definition at line 110 of file gnss_ephemeris.h.

12.164.3.28 satvel_Y

`double Gnss_Ephemeris::satvel_Y {}`
Earth-fixed velocity coordinate y of the satellite [m].
Definition at line 111 of file gnss_ephemeris.h.

12.164.3.29 satvel_Z

`double Gnss_Ephemeris::satvel_Z {}`
Earth-fixed velocity coordinate z of the satellite [m].
Definition at line 112 of file gnss_ephemeris.h.

12.164.3.30 sqrtA

`double Gnss_Ephemeris::sqrtA {}`
Square root of the semi-major axis [meters^1/2].
Definition at line 76 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.31 System

`char Gnss_Ephemeris::System {} [protected]`
Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.
Definition at line 115 of file gnss_ephemeris.h.
Referenced by Gps_CNAV_Ephemeris::Gps_CNAV_Ephemeris(), and Gps_Ephemeris::Gps_Ephemeris().

12.164.3.32 toc

`int32_t Gnss_Ephemeris::toc {}`
Clock correction data reference Time of Week [sec].
Definition at line 92 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.33 toe

`int32_t Gnss_Ephemeris::toe {}`
Ephemeris reference time [s].
Definition at line 89 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.34 tow

`int32_t Gnss_Ephemeris::tow {}`
Time of Week.

Definition at line 102 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().

12.164.3.35 WN

```
int32_t Gnss_Ephemeris::WN {}
```

Week number.
Definition at line 101 of file gnss_ephemeris.h.
Referenced by Serdes_Galileo_Eph::readProtobuffer(), Serdes_Gps_Eph::readProtobuffer(), Beidou_Dnav_Ephemeris::serialize(), Galileo_Ephemeris::serialize(), Gps_CNAV_Ephemeris::serialize(), and Gps_Ephemeris::serialize().
The documentation for this class was generated from the following file:

- gnss_ephemeris.h

12.165 Gnss_Satellite Class Reference

This class represents a GNSS satellite.
`#include <gnss_satellite.h>`

Public Member Functions

- Gnss_Satellite ()=default

  Default Constructor.
- Gnss_Satellite (const std::string &system_, uint32_t PRN_)

  Concrete GNSS satellite Constructor.
- ~Gnss_Satellite ()=default

  Default Destructor.
- Gnss_Satellite (const Gnss_Satellite &other) noexcept

  Copy constructor.
- Gnss_Satellite & operator= (const Gnss_Satellite &) noexcept

  Copy assignment operator.
- Gnss_Satellite (Gnss_Satellite &&other) noexcept

  Move constructor.
- Gnss_Satellite & operator= (Gnss_Satellite &&other) noexcept

  Move assignment operator.
- void update_PRN (uint32_t PRN)

  Updates the PRN Number (and the corresponding block) when the slot number is decoded, only applies to GLONASS GNAV messages.
- uint32_t get_PRN () const

  Gets satellite's PRN.
- int32_t get_rf_link () const

  Gets the satellite's rf link.
- std::string get_system () const

  Gets the satellite system {"GPS", "Glonass", "SBAS", "Galileo", "Beidou", "QZSS"}.
- std::string get_system_short () const

  Gets the satellite system {"G", "R", "SBAS", "E", "C", "J"}.
- std::string get_block () const

  Gets the satellite block. If GPS, returns {"IIA", "IIR", "IIR-M", "IIF"}.
- std::string what_block (const std::string &system_, uint32_t PRN_)

  Gets the block of a given satellite.

Friends

- bool operator== (const Gnss_Satellite &, const Gnss_Satellite &)

  operator== for comparison
- std::ostream & operator<< (std::ostream &, const Gnss_Satellite &)

  operator<< for pretty printing

12.165.1 Detailed Description

This class represents a GNSS satellite.
It contains information about the space vehicles currently operational of GPS, Glonass, SBAS and Galileo constellations.
Definition at line 39 of file gnss_satellite.h.

12.165.2 Constructor & Destructor Documentation

12.165.2.1 Gnss_Satellite() [1/4]

```
Gnss_Satellite::Gnss_Satellite ()  [default]
```

Default Constructor.
Referenced by Gnss_Satellite(), Gnss_Satellite(), operator<<, operator=(), operator=(), and operator==.

12.165.2.2 Gnss_Satellite() [2/4]

```
Gnss_Satellite::Gnss_Satellite (
            const std::string & system_,
            uint32_t PRN_)
```

Concrete GNSS satellite Constructor.

12.165.2.3 ~Gnss_Satellite()

```
Gnss_Satellite::~Gnss_Satellite ()  [default]
```

Default Destructor.

12.165.2.4 Gnss_Satellite() [3/4]

```
Gnss_Satellite::Gnss_Satellite (
            const Gnss_Satellite & other)  [noexcept]
```

Copy constructor.
References Gnss_Satellite().

12.165.2.5 Gnss_Satellite() [4/4]

```
Gnss_Satellite::Gnss_Satellite (
            Gnss_Satellite && other)  [noexcept]
```

Move constructor.
References Gnss_Satellite().

12.165.3 Member Function Documentation

12.165.3.1 get_block()

```
std::string Gnss_Satellite::get_block () const
```

Gets the satellite block. If GPS, returns {"IIA", "IIR", "IIR-M", "IIF"}.

12.165.3.2 get_PRN()

```
uint32_t Gnss_Satellite::get_PRN () const
```

Gets satellite's PRN.

12.165.3.3 get_rf_link()

```
int32_t Gnss_Satellite::get_rf_link () const
```
Gets the satellite's rf link.

12.165.3.4 get_system()

```
std::string Gnss_Satellite::get_system () const
```
Gets the satellite system {"GPS", "Glonass", "SBAS", "Galileo", "Beidou", "QZSS"}.

12.165.3.5 get_system_short()

```
std::string Gnss_Satellite::get_system_short () const
```
Gets the satellite system {"G", "R", "SBAS", "E", "C", "J"}.

12.165.3.6 operator=() [1/2]

```
Gnss_Satellite & Gnss_Satellite::operator= (
            const Gnss_Satellite & )  [noexcept]
```
Copy assignment operator.
References Gnss_Satellite().

12.165.3.7 operator=() [2/2]

```
Gnss_Satellite & Gnss_Satellite::operator= (
            Gnss_Satellite && other)  [noexcept]
```
Move assignment operator.
References Gnss_Satellite().

12.165.3.8 update_PRN()

```
void Gnss_Satellite::update_PRN (
            uint32_t PRN)
```
Updates the PRN Number (and the corresponding block) when the slot number is decoded, only applies to GLONASS GNAV messages.

12.165.3.9 what_block()

```
std::string Gnss_Satellite::what_block (
            const std::string & system_,
            uint32_t PRN_)
```
Gets the block of a given satellite.

12.165.4 Friends And Related Symbol Documentation

12.165.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & ,
            const Gnss_Satellite & )  [friend]
```
operator<< for pretty printing
References Gnss_Satellite().

12.165.4.2 operator==

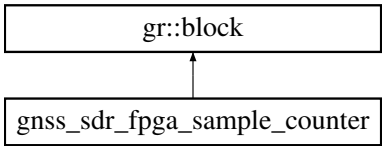
```
bool operator== (
            const Gnss_Satellite & ,
            const Gnss_Satellite & )  [friend]
```
operator== for comparison
References Gnss_Satellite().

The documentation for this class was generated from the following file:

- gnss_satellite.h

12.166 gnss_sdr_fpga_sample_counter Class Reference

Inheritance diagram for gnss_sdr_fpga_sample_counter:



Public Member Functions

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- gnss_sdr_fpga_sample_counter_sptr **gnss_sdr_make_fpga_sample_counter** (double _fs, int32_t _interval_ms)

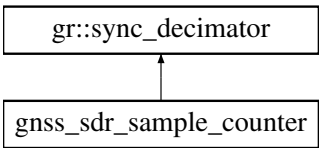
12.166.1 Detailed Description

Definition at line 40 of file gnss_sdr_fpga_sample_counter.h.
The documentation for this class was generated from the following file:

- gnss_sdr_fpga_sample_counter.h

12.167 gnss_sdr_sample_counter Class Reference

Inheritance diagram for gnss_sdr_sample_counter:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- gnss_sdr_sample_counter_sptr **gnss_sdr_make_sample_counter** (double _fs, int32_t _interval_ms, size_t _size)

12.167.1 Detailed Description

Definition at line 43 of file gnss_sdr_sample_counter.h.
The documentation for this class was generated from the following file:

- gnss_sdr_sample_counter.h

12.168 Gnss_Sdr_Supl_Client Class Reference

class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library..
`#include <gnss_sdr_supl_client.h>`

Public Member Functions

- int **get_assistance** (int i_mcc, int i_mns, int i_lac, int i_ci)
- void **read_supl_data** ()
- bool load_ephemeris_xml (const std::string &file_name)

  Read GPS NAV ephemeris map from XML file.
- bool save_ephemeris_map_xml (const std::string &file_name, std::map< int, Gps_Ephemeris > eph_map)

  Save ephemeris map to XML file.
- bool load_cnav_ephemeris_xml (const std::string &file_name)

  Read GPS CNAV ephemeris map from XML file.
- bool save_cnav_ephemeris_map_xml (const std::string &file_name, std::map< int, Gps_CNAV_Ephemeris > eph_map)

  Save GPS CNAV ephemeris map to XML file.
- bool load_gal_ephemeris_xml (const std::string &file_name)

  Read Galileo ephemeris map from XML file.
- bool save_gal_ephemeris_map_xml (const std::string &file_name, std::map< int, Galileo_Ephemeris > eph_map)

  Save Galileo ephemeris map to XML file.
- bool load_gnav_ephemeris_xml (const std::string &file_name)

  Read GLONASS GNAV ephemeris map from XML file.
- bool save_gnav_ephemeris_map_xml (const std::string &file_name, std::map< int, Glonass_Gnav_Ephemeris > eph_map)

  Save GLONASS GNAV ephemeris map to XML file.
- bool load_utc_xml (const std::string &file_name)

  Read GPS utc model from XML file.
- bool save_utc_xml (const std::string &file_name, Gps_Utc_Model &utc)

  Save UTC model map to XML file.
- bool load_cnav_utc_xml (const std::string &file_name)

  Read CNAV GPS utc model from XML file.
- bool save_cnav_utc_xml (const std::string &file_name, Gps_CNAV_Utc_Model &utc)

  Save CNAV UTC model map to XML file.
- bool load_gal_utc_xml (const std::string &file_name)

  Read Galileo utc model from XML file.
- bool save_gal_utc_xml (const std::string &file_name, Galileo_Utc_Model &utc)

  Save Galileo UTC model map to XML file.
- bool load_gal_almanac_xml (const std::string &file_name)

  Read Galileo almanac map from XML file.
- bool save_gal_almanac_xml (const std::string &file_name, std::map< int, Galileo_Almanac > galileo_↩ almanac_map_to_save)

  Save Galileo almanac map to XML file.
- bool load_gps_almanac_xml (const std::string &file_name)

  Read GPS almanac map from XML file.
- bool save_gps_almanac_xml (const std::string &file_name, std::map< int, Gps_Almanac > gps_almanac↩ _map_to_save)

  Save GPS almanac map to XML file.
- bool load_iono_xml (const std::string &file_name)

  Read iono from XML file.
- bool save_iono_xml (const std::string &file_name, Gps_Iono &iono)

Save iono map to XML file.

- bool load_gal_iono_xml (const std::string &file_name)

  Read Galileo iono from XML file.
- bool save_gal_iono_xml (const std::string &file_name, Galileo_Iono &iono)

  Save Galileo iono map to XML file.
- bool load_glo_utc_xml (const std::string &file_name)

  Read Glonass utc model from XML file.
- bool save_glo_utc_xml (const std::string &file_name, Glonass_Gnav_Utc_Model &utc)

  Save Glonass UTC model map to XML file.
- bool load_ref_time_xml (const std::string &file_name)

  Read ref time from XML file.
- bool save_ref_time_xml (const std::string &file_name, Agnss_Ref_Time &ref_time_map)

  Save ref time map to XML file.
- bool load_ref_location_xml (const std::string &file_name)

  Read ref location from XML file.
- bool save_ref_location_xml (const std::string &file_name, Agnss_Ref_Location &ref_location)

  Save ref location map to XML file.
- void **print_assistance** ()

Public Attributes

- std::string server_name
- int server_port
- int request
- std::map< int, Gps_Ephemeris > gps_ephemeris_map
- std::map< int, Galileo_Ephemeris > gal_ephemeris_map
- std::map< int, Gps_CNAV_Ephemeris > gps_cnav_ephemeris_map
- std::map< int, Glonass_Gnav_Ephemeris > glonass_gnav_ephemeris_map
- std::map< int, Gps_Almanac > gps_almanac_map
- std::map< int, Galileo_Almanac > gal_almanac_map
- Gps_Iono gps_iono
- Galileo_Iono gal_iono
- Agnss_Ref_Time gps_time
- Gps_Utc_Model gps_utc
- Galileo_Utc_Model gal_utc
- Gps_CNAV_Utc_Model gps_cnav_utc
- Glonass_Gnav_Utc_Model glo_gnav_utc
- Agnss_Ref_Location gps_ref_loc
- std::map< int, Gps_Acq_Assist > gps_acq_map

12.168.1 Detailed Description

class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library..
Definition at line 56 of file gnss_sdr_supl_client.h.

12.168.2 Member Function Documentation

12.168.2.1 load_cnav_ephemeris_xml()

```
bool Gnss_Sdr_Supl_Client::load_cnav_ephemeris_xml (
            const std::string & file_name)
```

Read GPS CNAV ephemeris map from XML file.

12.168.2.2 load_cnav_utc_xml()

```
bool Gnss_Sdr_Supl_Client::load_cnav_utc_xml (
            const std::string & file_name)
```

Read CNAV GPS utc model from XML file.

12.168.2.3 load_ephemeris_xml()

```
bool Gnss_Sdr_Supl_Client::load_ephemeris_xml (
            const std::string & file_name)
```

Read GPS NAV ephemeris map from XML file.

12.168.2.4 load_gal_almanac_xml()

```
bool Gnss_Sdr_Supl_Client::load_gal_almanac_xml (
            const std::string & file_name)
```

Read Galileo almanac map from XML file.

12.168.2.5 load_gal_ephemeris_xml()

```
bool Gnss_Sdr_Supl_Client::load_gal_ephemeris_xml (
            const std::string & file_name)
```

Read Galileo ephemeris map from XML file.

12.168.2.6 load_gal_iono_xml()

```
bool Gnss_Sdr_Supl_Client::load_gal_iono_xml (
            const std::string & file_name)
```

Read Galileo iono from XML file.

12.168.2.7 load_gal_utc_xml()

```
bool Gnss_Sdr_Supl_Client::load_gal_utc_xml (
            const std::string & file_name)
```

Read Galileo utc model from XML file.

12.168.2.8 load_glo_utc_xml()

```
bool Gnss_Sdr_Supl_Client::load_glo_utc_xml (
            const std::string & file_name)
```

Read Glonass utc model from XML file.

12.168.2.9 load_gnav_ephemeris_xml()

```
bool Gnss_Sdr_Supl_Client::load_gnav_ephemeris_xml (
            const std::string & file_name)
```

Read GLONASS GNAV ephemeris map from XML file.

12.168.2.10 load_gps_almanac_xml()

```
bool Gnss_Sdr_Supl_Client::load_gps_almanac_xml (
            const std::string & file_name)
```

Read GPS almanac map from XML file.

12.168.2.11 load_iono_xml()

```
bool Gnss_Sdr_Supl_Client::load_iono_xml (
            const std::string & file_name)
```

Read iono from XML file.

12.168.2.12 load_ref_location_xml()

```
bool Gnss_Sdr_Supl_Client::load_ref_location_xml (
            const std::string & file_name)
```

Read ref location from XML file.

12.168.2.13 load_ref_time_xml()

```
bool Gnss_Sdr_Supl_Client::load_ref_time_xml (
            const std::string & file_name)
```

Read ref time from XML file.

12.168.2.14 load_utc_xml()

```
bool Gnss_Sdr_Supl_Client::load_utc_xml (
            const std::string & file_name)
```

Read GPS utc model from XML file.

12.168.2.15 save_cnav_ephemeris_map_xml()

```
bool Gnss_Sdr_Supl_Client::save_cnav_ephemeris_map_xml (
            const std::string & file_name,
            std::map< int, Gps_CNAV_Ephemeris > eph_map)
```

Save GPS CNAV ephemeris map to XML file.

12.168.2.16 save_cnav_utc_xml()

```
bool Gnss_Sdr_Supl_Client::save_cnav_utc_xml (
            const std::string & file_name,
            Gps_CNAV_Utc_Model & utc)
```

Save CNAV UTC model map to XML file.

12.168.2.17 save_ephemeris_map_xml()

```
bool Gnss_Sdr_Supl_Client::save_ephemeris_map_xml (
            const std::string & file_name,
            std::map< int, Gps_Ephemeris > eph_map)
```

Save ephemeris map to XML file.

12.168.2.18 save_gal_almanac_xml()

```
bool Gnss_Sdr_Supl_Client::save_gal_almanac_xml (
            const std::string & file_name,
            std::map< int, Galileo_Almanac > galileo_almanac_map_to_save)
```

Save Galileo almanac map to XML file.

12.168.2.19 save_gal_ephemeris_map_xml()

```
bool Gnss_Sdr_Supl_Client::save_gal_ephemeris_map_xml (
            const std::string & file_name,
            std::map< int, Galileo_Ephemeris > eph_map)
```

Save Galileo ephemeris map to XML file.

12.168.2.20 save_gal_iono_xml()

```
bool Gnss_Sdr_Supl_Client::save_gal_iono_xml (
            const std::string & file_name,
            Galileo_Iono & iono)
```

Save Galileo iono map to XML file.

12.168.2.21 save_gal_utc_xml()

```
bool Gnss_Sdr_Supl_Client::save_gal_utc_xml (
            const std::string & file_name,
            Galileo_Utc_Model & utc)
```

Save Galileo UTC model map to XML file.

12.168.2.22 save_glo_utc_xml()

```
bool Gnss_Sdr_Supl_Client::save_glo_utc_xml (
            const std::string & file_name,
            Glonass_Gnav_Utc_Model & utc)
```

Save Glonass UTC model map to XML file.

12.168.2.23 save_gnav_ephemeris_map_xml()

```
bool Gnss_Sdr_Supl_Client::save_gnav_ephemeris_map_xml (
            const std::string & file_name,
            std::map< int, Glonass_Gnav_Ephemeris > eph_map)
```

Save GLONASS GNAV ephemeris map to XML file.

12.168.2.24 save_gps_almanac_xml()

```
bool Gnss_Sdr_Supl_Client::save_gps_almanac_xml (
            const std::string & file_name,
            std::map< int, Gps_Almanac > gps_almanac_map_to_save)
```

Save GPS almanac map to XML file.

12.168.2.25 save_iono_xml()

```
bool Gnss_Sdr_Supl_Client::save_iono_xml (
            const std::string & file_name,
            Gps_Iono & iono)
```

Save iono map to XML file.

12.168.2.26 save_ref_location_xml()

```
bool Gnss_Sdr_Supl_Client::save_ref_location_xml (
            const std::string & file_name,
            Agnss_Ref_Location & ref_location)
```

Save ref location map to XML file.

12.168.2.27 save_ref_time_xml()

```
bool Gnss_Sdr_Supl_Client::save_ref_time_xml (
            const std::string & file_name,
            Agnss_Ref_Time & ref_time_map)
```

Save ref time map to XML file.

12.168.2.28 save_utc_xml()

```
bool Gnss_Sdr_Supl_Client::save_utc_xml (
            const std::string & file_name,
            Gps_Utc_Model & utc)
```

Save UTC model map to XML file.

12.168.3 Member Data Documentation

12.168.3.1 gal_almanac_map

`std::map<int, Galileo_Almanac> Gnss_Sdr_Supl_Client::gal_almanac_map`
Definition at line 74 of file gnss_sdr_supl_client.h.

12.168.3.2 gal_ephemeris_map

`std::map<int, Galileo_Ephemeris> Gnss_Sdr_Supl_Client::gal_ephemeris_map`
Definition at line 68 of file gnss_sdr_supl_client.h.

12.168.3.3 gal_iono

`Galileo_Iono Gnss_Sdr_Supl_Client::gal_iono`
Definition at line 78 of file gnss_sdr_supl_client.h.

12.168.3.4 gal_utc

`Galileo_Utc_Model Gnss_Sdr_Supl_Client::gal_utc`
Definition at line 83 of file gnss_sdr_supl_client.h.

12.168.3.5 glo_gnav_utc

`Glonass_Gnav_Utc_Model Gnss_Sdr_Supl_Client::glo_gnav_utc`
Definition at line 85 of file gnss_sdr_supl_client.h.

12.168.3.6 glonass_gnav_ephemeris_map

`std::map<int, Glonass_Gnav_Ephemeris> Gnss_Sdr_Supl_Client::glonass_gnav_ephemeris_map`
Definition at line 70 of file gnss_sdr_supl_client.h.

12.168.3.7 gps_acq_map

`std::map<int, Gps_Acq_Assist> Gnss_Sdr_Supl_Client::gps_acq_map`
Definition at line 89 of file gnss_sdr_supl_client.h.

12.168.3.8 gps_almanac_map

`std::map<int, Gps_Almanac> Gnss_Sdr_Supl_Client::gps_almanac_map`
Definition at line 73 of file gnss_sdr_supl_client.h.

12.168.3.9 gps_cnav_ephemeris_map

`std::map<int, Gps_CNAV_Ephemeris> Gnss_Sdr_Supl_Client::gps_cnav_ephemeris_map`
Definition at line 69 of file gnss_sdr_supl_client.h.

12.168.3.10 gps_cnav_utc

`Gps_CNAV_Utc_Model Gnss_Sdr_Supl_Client::gps_cnav_utc`
Definition at line 84 of file gnss_sdr_supl_client.h.

12.168.3.11 gps_ephemeris_map

`std::map<int, Gps_Ephemeris> Gnss_Sdr_Supl_Client::gps_ephemeris_map`
Definition at line 67 of file gnss_sdr_supl_client.h.

12.168.3.12 gps_iono

`Gps_Iono Gnss_Sdr_Supl_Client::gps_iono`
Definition at line 77 of file gnss_sdr_supl_client.h.

12.168.3.13 gps_ref_loc

`Agnss_Ref_Location Gnss_Sdr_Supl_Client::gps_ref_loc`
Definition at line 87 of file gnss_sdr_supl_client.h.

12.168.3.14 gps_time

`Agnss_Ref_Time Gnss_Sdr_Supl_Client::gps_time`
Definition at line 80 of file gnss_sdr_supl_client.h.

12.168.3.15 gps_utc

`Gps_Utc_Model Gnss_Sdr_Supl_Client::gps_utc`
Definition at line 82 of file gnss_sdr_supl_client.h.

12.168.3.16 request

`int Gnss_Sdr_Supl_Client::request`
Definition at line 65 of file gnss_sdr_supl_client.h.

12.168.3.17 server_name

`std::string Gnss_Sdr_Supl_Client::server_name`
Definition at line 63 of file gnss_sdr_supl_client.h.

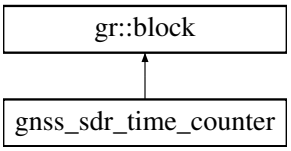
12.168.3.18 server_port

`int Gnss_Sdr_Supl_Client::server_port`
Definition at line 64 of file gnss_sdr_supl_client.h.
The documentation for this class was generated from the following file:

- gnss_sdr_supl_client.h

12.169 gnss_sdr_time_counter Class Reference

Inheritance diagram for gnss_sdr_time_counter:



Public Member Functions

- int **general_work** (int noutput_items __attribute__((unused)), gr_vector_int &ninput_items __attribute↩ __((unused)), gr_vector_const_void_star &input_items __attribute__((unused)), gr_vector_void_star &output_items)

Friends

- gnss_sdr_time_counter_sptr **gnss_sdr_make_time_counter** ()

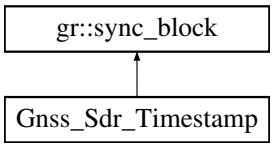
12.169.1 Detailed Description

Definition at line 38 of file gnss_sdr_time_counter.h.
The documentation for this class was generated from the following file:

- gnss_sdr_time_counter.h

12.170 Gnss_Sdr_Timestamp Class Reference

Inheritance diagram for Gnss_Sdr_Timestamp:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- bool **start** ()

Friends

- gnss_shared_ptr< Gnss_Sdr_Timestamp > **gnss_sdr_make_Timestamp** (size_t sizeof_stream_item, std::string timestamp_file, double clock_offset_ms, int items_to_samples)

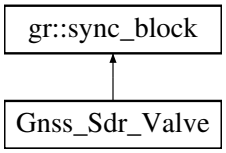
12.170.1 Detailed Description

Definition at line 46 of file gnss_sdr_timestamp.h.
The documentation for this class was generated from the following file:

- gnss_sdr_timestamp.h

12.171 Gnss_Sdr_Valve Class Reference

Implementation of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.
`#include <gnss_sdr_valve.h>`
Inheritance diagram for Gnss_Sdr_Valve:



Public Member Functions

- void **open_valve** ()
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue, bool stop_flowgraph)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue, bool stop_flowgraph, int n_streams)

12.171.1 Detailed Description

Implementation of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.
Definition at line 62 of file gnss_sdr_valve.h.
The documentation for this class was generated from the following file:

- gnss_sdr_valve.h

12.172 Gnss_Signal Class Reference

This class represents a GNSS signal.
`#include <gnss_signal.h>`

Public Member Functions

- **Gnss_Signal** (const std::string &signal_)
- **Gnss_Signal** (const Gnss_Satellite &satellite_, const std::string &signal_)
- std::string get_signal_str () const

  Return the signal identifier string.
- Gnss_Satellite get_satellite () const

  Get the Gnss_Satellite associated to the signal.

Friends

- bool operator== (const Gnss_Signal &, const Gnss_Signal &)

  operator== for comparison
- std::ostream & operator<< (std::ostream &, const Gnss_Signal &)

  operator<< for pretty printing

12.172.1 Detailed Description

This class represents a GNSS signal.
Encapsulates a specific GNSS signal (e.g., GPS L1 C/A, Galileo E1B) and its associated satellite.
Definition at line 38 of file gnss_signal.h.

12.172.2 Member Function Documentation

12.172.2.1 get_satellite()

`Gnss_Satellite Gnss_Signal::get_satellite () const`
Get the Gnss_Satellite associated to the signal.

12.172.2.2 get_signal_str()

`std::string Gnss_Signal::get_signal_str () const`
Return the signal identifier string.

- GPS: "1C" (L1 C/A), "2S" (L2C), "L5"
- GLONASS: "1G" (L1 C/A), "2G" (L2 C/A)
- Galileo: "1B" (E1B), "5X" (E5a), "7X" (E5b), "6C" (E6C)
- BeiDou: "B1" (B1I), "B3" (B3I), "1D" (B1C), "5D" (B2a data)

12.172.3 Friends And Related Symbol Documentation

12.172.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & ,
            const Gnss_Signal & )  [friend]
```

operator<< for pretty printing

12.172.3.2 operator==

```
bool operator== (
            const Gnss_Signal & ,
            const Gnss_Signal & )  [friend]
```

operator== for comparison

The documentation for this class was generated from the following file:

- gnss_signal.h

12.173 Gnss_Synchro Class Reference

This is the class that contains the information that is shared by the processing blocks.

`#include <gnss_synchro.h>`

Public Member Functions

- Gnss_Synchro ()=default

  Default constructor.
- ~Gnss_Synchro ()=default

  Default destructor.
- Gnss_Synchro (const Gnss_Synchro &other) noexcept=default

  Copy constructor.
- Gnss_Synchro & operator= (const Gnss_Synchro &rhs) noexcept

  Copy assignment operator.
- Gnss_Synchro (Gnss_Synchro &&other) noexcept=default

  Move constructor.
- Gnss_Synchro & operator= (Gnss_Synchro &&other) noexcept

  Move assignment operator.
- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

  This member function serializes and restores Gnss_Synchro objects from a byte stream.

Public Attributes

- char System {}

  Set by Channel::set_signal(Gnss_Signal gnss_signal).
- char Signal [3] {}

  Set by Channel::set_signal(Gnss_Signal gnss_signal).
- uint32_t PRN {}

  Set by Channel::set_signal(Gnss_Signal gnss_signal).
- int32_t Channel_ID {}

  Set by Channel constructor.
- double Acq_delay_samples {}

  Set by Acquisition processing block.
- double Acq_doppler_hz {}

Set by Acquisition processing block.

- uint64_t Acq_samplestamp_samples {}

  Set by Acquisition processing block.

- uint32_t Acq_doppler_step {}

  Set by Acquisition processing block.

- int64_t fs {}

  Set by Tracking processing block.

- double Prompt_I {}

  Set by Tracking processing block.

- double Prompt_Q {}

  Set by Tracking processing block.

- double CN0_dB_hz {}

  Set by Tracking processing block.

- double Carrier_Doppler_hz {}

  Set by Tracking processing block.

- double Carrier_phase_rads {}

  Set by Tracking processing block.

- double Code_phase_samples {}

  Set by Tracking processing block.

- uint64_t Tracking_sample_counter {}

  Set by Tracking processing block.

- int32_t correlation_length_ms {}

  Set by Tracking processing block.

- uint32_t TOW_at_current_symbol_ms {}

  Set by Telemetry Decoder processing block.

- double Pseudorange_m {}

  Set by Observables processing block.

- double RX_time {}

  Set by Observables processing block.

- double interp_TOW_ms {}

  Set by Observables processing block.

- bool Flag_valid_acquisition {}

  Set by Acquisition processing block.

- bool Flag_valid_symbol_output {}

  Set by Tracking processing block.

- bool Flag_valid_word {}

  Set by Telemetry Decoder processing block.

- bool Flag_valid_pseudorange {}

  Set by Observables processing block.

- bool Flag_PLL_180_deg_phase_locked {}

  Set by Telemetry Decoder processing block.

- bool Flag_cycle_slip {}

  Set by Observables processing block.

- bool Flag_half_cycle_slip {}

  Set by Observables processing block. Signals that the carrier phase has jumped by half a cycle because the Costas loop phase ambiguity resolved by the Telemetry Decoder changed.

- bool Flag_carrier_phase_continuous {true}

  Set by Tracking processing block. False on the first output after the accumulated carrier phase has been (re)initialized, that is, whenever the carrier phase ambiguity of the channel has changed.

12.173.1 Detailed Description

This is the class that contains the information that is shared by the processing blocks.
Definition at line 38 of file gnss_synchro.h.

12.173.2 Constructor & Destructor Documentation

12.173.2.1 Gnss_Synchro() [1/3]

```
Gnss_Synchro::Gnss_Synchro ()  [default]
```

Default constructor.
Referenced by Gnss_Synchro(), Gnss_Synchro(), operator=(), operator=(), and Serdes_Gnss_Synchro::readProtobuffer().

12.173.2.2 ~Gnss_Synchro()

```
Gnss_Synchro::~Gnss_Synchro ()  [default]
```

Default destructor.

12.173.2.3 Gnss_Synchro() [2/3]

```
Gnss_Synchro::Gnss_Synchro (
            const Gnss_Synchro & other)  [default], [noexcept]
```

Copy constructor.
References Gnss_Synchro().

12.173.2.4 Gnss_Synchro() [3/3]

```
Gnss_Synchro::Gnss_Synchro (
            Gnss_Synchro && other)  [default], [noexcept]
```

Move constructor.
References Gnss_Synchro().

12.173.3 Member Function Documentation

12.173.3.1 operator=() [1/2]

```
Gnss_Synchro & Gnss_Synchro::operator= (
            const Gnss_Synchro & rhs)  [inline], [noexcept]
```

Copy assignment operator.
Definition at line 98 of file gnss_synchro.h.
References Gnss_Synchro().

12.173.3.2 operator=() [2/2]

```
Gnss_Synchro & Gnss_Synchro::operator= (
            Gnss_Synchro && other)  [inline], [noexcept]
```

Move assignment operator.
Definition at line 142 of file gnss_synchro.h.
References Gnss_Synchro().

12.173.3.3 serialize()

```
template<class Archive>
void Gnss_Synchro::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

This member function serializes and restores Gnss_Synchro objects from a byte stream.
Definition at line 220 of file gnss_synchro.h.
References Acq_delay_samples, Acq_doppler_hz, Acq_doppler_step, Acq_samplestamp_samples, Carrier_Doppler_hz, Carrier_phase_rads, Channel_ID, CN0_dB_hz, Code_phase_samples, correlation_length_ms, Flag_carrier_phase_continuous,

Flag_cycle_slip, Flag_half_cycle_slip, Flag_PLL_180_deg_phase_locked, Flag_valid_acquisition, Flag_valid_pseudorange, Flag_valid_symbol_output, Flag_valid_word, fs, interp_TOW_ms, PRN, Prompt_I, Prompt_Q, Pseudorange_m, RX_time, Signal, System, TOW_at_current_symbol_ms, and Tracking_sample_counter.

12.173.4 Member Data Documentation

12.173.4.1 Acq_delay_samples

```
double Gnss_Synchro::Acq_delay_samples {}
```

Set by Acquisition processing block.
Definition at line 52 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.2 Acq_doppler_hz

```
double Gnss_Synchro::Acq_doppler_hz {}
```

Set by Acquisition processing block.
Definition at line 53 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.3 Acq_doppler_step

```
uint32_t Gnss_Synchro::Acq_doppler_step {}
```

Set by Acquisition processing block.
Definition at line 55 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.4 Acq_samplestamp_samples

```
uint64_t Gnss_Synchro::Acq_samplestamp_samples {}
```

Set by Acquisition processing block.
Definition at line 54 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.5 Carrier_Doppler_hz

```
double Gnss_Synchro::Carrier_Doppler_hz {}
```

Set by Tracking processing block.
Definition at line 62 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.6 Carrier_phase_rads

```
double Gnss_Synchro::Carrier_phase_rads {}
```

Set by Tracking processing block.
Definition at line 63 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.7 Channel_ID

```
int32_t Gnss_Synchro::Channel_ID {}
```

Set by Channel constructor.
Definition at line 49 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.8 CN0_dB_hz

```
double Gnss_Synchro::CN0_dB_hz {}
```

Set by Tracking processing block.
Definition at line 61 of file gnss_synchro.h.

Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.9 Code_phase_samples

```
double Gnss_Synchro::Code_phase_samples {}
```
Set by Tracking processing block.
Definition at line 64 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.10 correlation_length_ms

```
int32_t Gnss_Synchro::correlation_length_ms {}
```
Set by Tracking processing block.
Definition at line 66 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.11 Flag_carrier_phase_continuous

```
bool Gnss_Synchro::Flag_carrier_phase_continuous {true}
```
Set by Tracking processing block. False on the first output after the accumulated carrier phase has been (re)initialized, that is, whenever the carrier phase ambiguity of the channel has changed.
Definition at line 92 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.12 Flag_cycle_slip

```
bool Gnss_Synchro::Flag_cycle_slip {}
```
Set by Observables processing block.
Definition at line 82 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.13 Flag_half_cycle_slip

```
bool Gnss_Synchro::Flag_half_cycle_slip {}
```
Set by Observables processing block. Signals that the carrier phase has jumped by half a cycle because the Costas loop phase ambiguity resolved by the Telemetry Decoder changed.
Definition at line 87 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.14 Flag_PLL_180_deg_phase_locked

```
bool Gnss_Synchro::Flag_PLL_180_deg_phase_locked {}
```
Set by Telemetry Decoder processing block.
Definition at line 81 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.15 Flag_valid_acquisition

```
bool Gnss_Synchro::Flag_valid_acquisition {}
```
Set by Acquisition processing block.
Definition at line 77 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.16 Flag_valid_pseudorange

```
bool Gnss_Synchro::Flag_valid_pseudorange {}
```
Set by Observables processing block.
Definition at line 80 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.17 Flag_valid_symbol_output

```
bool Gnss_Synchro::Flag_valid_symbol_output {}
```
Set by Tracking processing block.
Definition at line 78 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.18 Flag_valid_word

```
bool Gnss_Synchro::Flag_valid_word {}
```
Set by Telemetry Decoder processing block.
Definition at line 79 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.19 fs

```
int64_t Gnss_Synchro::fs {}
```
Set by Tracking processing block.
Definition at line 58 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.20 interp_TOW_ms

```
double Gnss_Synchro::interp_TOW_ms {}
```
Set by Observables processing block.
Definition at line 74 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.21 PRN

```
uint32_t Gnss_Synchro::PRN {}
```
Set by Channel::set_signal(Gnss_Signal gnss_signal).
Definition at line 48 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.22 Prompt_I

```
double Gnss_Synchro::Prompt_I {}
```
Set by Tracking processing block.
Definition at line 59 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.23 Prompt_Q

```
double Gnss_Synchro::Prompt_Q {}
```
Set by Tracking processing block.
Definition at line 60 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.24 Pseudorange_m

```
double Gnss_Synchro::Pseudorange_m {}
```
Set by Observables processing block.
Definition at line 72 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.25 RX_time

```
double Gnss_Synchro::RX_time {}
```
Set by Observables processing block.

Definition at line 73 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.26 Signal

`char Gnss_Synchro::Signal[3] {}`
Set by Channel::set_signal(Gnss_Signal gnss_signal).
Definition at line 47 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.27 System

`char Gnss_Synchro::System {}`
Set by Channel::set_signal(Gnss_Signal gnss_signal).
Definition at line 46 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

12.173.4.28 TOW_at_current_symbol_ms

`uint32_t Gnss_Synchro::TOW_at_current_symbol_ms {}`
Set by Telemetry Decoder processing block.
Definition at line 69 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().

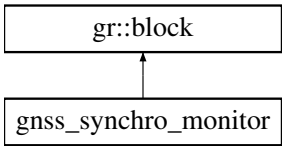
12.173.4.29 Tracking_sample_counter

`uint64_t Gnss_Synchro::Tracking_sample_counter {}`
Set by Tracking processing block.
Definition at line 65 of file gnss_synchro.h.
Referenced by Serdes_Gnss_Synchro::readProtobuffer(), and serialize().
The documentation for this class was generated from the following file:

- gnss_synchro.h

12.174 gnss_synchro_monitor Class Reference

This class implements a monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.
`#include <gnss_synchro_monitor.h>`
Inheritance diagram for gnss_synchro_monitor:



Public Member Functions

- ~gnss_synchro_monitor ()=default
  Default destructor.
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- gnss_synchro_monitor_sptr **gnss_synchro_make_monitor** (int n_channels, int decimation_factor, const std::vector< std::string > &udp_ports, const std::vector< std::string > &udp_addresses, bool enable_↩ protobuf)

12.174.1 Detailed Description

This class implements a monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.
Definition at line 53 of file gnss_synchro_monitor.h.

12.174.2 Constructor & Destructor Documentation

12.174.2.1 ~gnss_synchro_monitor()

```
gnss_synchro_monitor::~gnss_synchro_monitor ()  [default]
```

Default destructor.
The documentation for this class was generated from the following file:

- gnss_synchro_monitor.h

12.175 Gnss_Synchro_Udp_Sink Class Reference

This class sends serialized Gnss_Synchro objects over UDP to one or multiple endpoints.
`#include <gnss_synchro_udp_sink.h>`

Public Member Functions

- **Gnss_Synchro_Udp_Sink** (const std::vector< std::string > &addresses, const std::vector< std::string > &ports, bool enable_protobuf)
- bool **write_gnss_synchro** (const std::vector< Gnss_Synchro > &stocks)

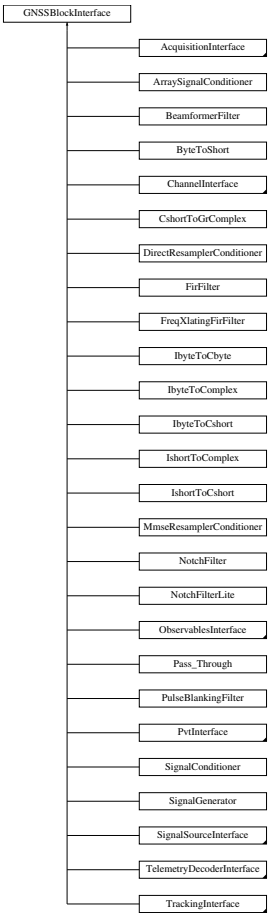
12.175.1 Detailed Description

This class sends serialized Gnss_Synchro objects over UDP to one or multiple endpoints.
Definition at line 45 of file gnss_synchro_udp_sink.h.
The documentation for this class was generated from the following file:

- gnss_synchro_udp_sink.h

12.176 GNSSBlockInterface Class Reference

This abstract class represents an interface to GNSS blocks.
`#include <gnss_block_interface.h>`
Inheritance diagram for GNSSBlockInterface:



Public Member Functions

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.176.1 Detailed Description

This abstract class represents an interface to GNSS blocks.
Abstract class for GNSS block interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 68 of file gnss_block_interface.h.

12.176.2 Member Function Documentation

12.176.2.1 connect()

```
virtual void GNSSBlockInterface::connect (
            gr::top_block_sptr top_block)  [pure virtual]
```

Implemented in Channel, DllPllTrackingAdapter, and FifoSignalSource.

12.176.2.2 disconnect()

```
virtual void GNSSBlockInterface::disconnect (
            gr::top_block_sptr top_block)  [pure virtual]
```

Implemented in DllPllTrackingAdapter.

12.176.2.3 get_left_block() [1/2]

```
virtual gr::basic_block_sptr GNSSBlockInterface::get_left_block ()  [pure virtual]
```

Implemented in DllPllTrackingAdapter.

12.176.2.4 get_left_block() [2/2]

```
virtual gr::basic_block_sptr GNSSBlockInterface::get_left_block (
            int RF_channel)  [inline], [virtual]
```

Definition at line 81 of file gnss_block_interface.h.

12.176.2.5 get_right_block() [1/2]

```
virtual gr::basic_block_sptr GNSSBlockInterface::get_right_block ()  [pure virtual]
```

Implemented in Channel, and DllPllTrackingAdapter.

12.176.2.6 get_right_block() [2/2]

```
virtual gr::basic_block_sptr GNSSBlockInterface::get_right_block (
            int RF_channel)  [inline], [virtual]
```

Definition at line 89 of file gnss_block_interface.h.

12.176.2.7 implementation()

```
virtual std::string GNSSBlockInterface::implementation ()  [pure virtual]
```

Implemented in ArraySignalConditioner, BeamformerFilter, ByteToShort, Channel, CshortToGrComplex, DirectResamplerConditioner, DllPllTrackingAdapter, FirFilter, FreqXlatingFirFilter, GalileoE1TcpConnectorTracking, GenSignalSource, GpsL1CaDllPllTrackingGPU, GpsL1CaGaussianTracking, GpsL1CaKfTracking, GpsL1CaTcpConnectorTracking, HybridObservables, IbyteToCbyte, IbyteToComplex, IbyteToCshort, IshortToComplex, IshortToCshort, NotchFilter, NotchFilterLite, Pass_Through, PulseBlankingFilter, Rtklib_Pvt, SignalConditioner, and SignalGenerator.

12.176.2.8 item_size()

```
virtual size_t GNSSBlockInterface::item_size ()  [pure virtual]
```

Implemented in DllPllTrackingAdapter, FileSourceBase, HybridObservables, and Rtklib_Pvt.

12.176.2.9 role()

```
virtual std::string GNSSBlockInterface::role ()  [pure virtual]
```

Implemented in DllPllTrackingAdapter.

12.176.2.10 start()

```
virtual void GNSSBlockInterface::start ()  [inline], [virtual]
```

Start the flow of samples if needed.
Reimplemented in DMASignalSourceFPGA.
Definition at line 101 of file gnss_block_interface.h.
The documentation for this class was generated from the following file:

- gnss_block_interface.h

12.177 GNSSFlowgraph Class Reference

This class represents a GNSS flow graph.
`#include <gnss_flowgraph.h>`

Public Member Functions

- GNSSFlowgraph (std::shared_ptr< ConfigurationInterface > configuration, std::shared_ptr< Concurrent_Queue< pmt::pmt_t > > queue)

  Constructor that initializes the receiver flow graph.
- ~GNSSFlowgraph ()

  Destructor.
- void start ()

  Start the flow graph.
- void stop ()

  Stop the flow graph.
- void connect ()

  Connects the defined blocks in the flow graph.
- void disconnect ()

  Disconnect the blocks in the flow graph.
- void wait ()

  Wait for a flowgraph to complete.
- void acquisition_manager (unsigned int who)

  Manage satellite acquisition.
- void apply_action (unsigned int who, unsigned int what)

  Applies an action to the flow graph.
- void set_configuration (const std::shared_ptr< ConfigurationInterface > &configuration)

  Set flow graph configuratiob.
- bool connected () const
- bool running () const
- bool send_telemetry_msg (const pmt::pmt_t &msg)

  Sends a GNU Radio asynchronous message from telemetry to PVT.
- std::shared_ptr< PvtInterface > get_pvt ()

  Returns a smart pointer to the PVT object.
- void priorize_satellites (const std::vector< std::pair< int, Gnss_Satellite > > &visible_satellites)

  Prioritize visible satellites in the specified vector.
- void MaybeUpdateVisibility ()

  Re-evaluates satellite visibility when warranted (new fix, new ephemeris/almanac, or recompute interval elapsed) so that subsequent search_next_signal() calls favor visible satellites. No-op unless GNSS-SDR.enable_visibility_↩ aware_search=true.
- void UpdateVisibilityReference (time_t utc_time, const std::array< float, 3 > &LLH)

  Sets the telecommand position/time reference and immediately refreshes visibility. Called by the control thread.
- bool visibility_aware_search_enabled () const

  Whether GNSS-SDR.enable_visibility_aware_search is on, so callers can skip the legacy get_visible_sats()/↩ priorize_satellites() startup reorder that MaybeUpdateVisibility() supersedes.
- void stop_duplicated_satellite_channels ()

  Stops any channel whose decoded satellite is already tracked, on the same signal, by another channel with higher C/N0. The stopped channel's assignment returns to the search pool and the channel is re-dispatched.

12.177.1 Detailed Description

This class represents a GNSS flow graph.
It contains a signal source, a signal conditioner, a set of channels, a PVT and an output filter.
Definition at line 72 of file gnss_flowgraph.h.

12.177.2 Constructor & Destructor Documentation

12.177.2.1 GNSSFlowgraph()

```
GNSSFlowgraph::GNSSFlowgraph (
            std::shared_ptr< ConfigurationInterface > configuration,
            std::shared_ptr< Concurrent_Queue< pmt::pmt_t > > queue)
```

Constructor that initializes the receiver flow graph.

12.177.2.2 ~GNSSFlowgraph()

```
GNSSFlowgraph::~GNSSFlowgraph ()
```

Destructor.

12.177.3 Member Function Documentation

12.177.3.1 acquisition_manager()

```
void GNSSFlowgraph::acquisition_manager (
            unsigned int who)
```

Manage satellite acquisition.

Parameters

in	*who*	Channel ID

12.177.3.2 apply_action()

```
void GNSSFlowgraph::apply_action (
            unsigned int who,
            unsigned int what)
```

Applies an action to the flow graph.

Parameters

in	*who*	Who generated the action
in	*what*	What is the action. 0: acquisition failed; 1: acquisition success; 2: tracking lost

12.177.3.3 connect()

```
void GNSSFlowgraph::connect ()
```

Connects the defined blocks in the flow graph.
Signal Source > Signal conditioner > Channels >> Observables >> PVT > Output filter

12.177.3.4 connected()

```
bool GNSSFlowgraph::connected () const  [inline]
```

Definition at line 136 of file gnss_flowgraph.h.

12.177.3.5 disconnect()

```
void GNSSFlowgraph::disconnect ()
```

Disconnect the blocks in the flow graph.

12.177.3.6 get_pvt()

```
std::shared_ptr< PvtInterface > GNSSFlowgraph::get_pvt ()  [inline]
```
Returns a smart pointer to the PVT object.
Definition at line 156 of file gnss_flowgraph.h.

12.177.3.7 MaybeUpdateVisibility()

```
void GNSSFlowgraph::MaybeUpdateVisibility ()
```
Re-evaluates satellite visibility when warranted (new fix, new ephemeris/almanac, or recompute interval elapsed) so that subsequent search_next_signal() calls favor visible satellites. No-op unless GNSS-SDR.enable_visibility↩ _aware_search=true.

12.177.3.8 priorize_satellites()

```
void GNSSFlowgraph::priorize_satellites (
            const std::vector< std::pair< int, Gnss_Satellite > > & visible_satellites)
```
Prioritize visible satellites in the specified vector.

12.177.3.9 running()

```
bool GNSSFlowgraph::running () const  [inline]
```
Definition at line 141 of file gnss_flowgraph.h.

12.177.3.10 send_telemetry_msg()

```
bool GNSSFlowgraph::send_telemetry_msg (
            const pmt::pmt_t & msg)
```
Sends a GNU Radio asynchronous message from telemetry to PVT.
It is used to assist the receiver with external ephemeris data

12.177.3.11 set_configuration()

```
void GNSSFlowgraph::set_configuration (
            const std::shared_ptr< ConfigurationInterface > & configuration)
```
Set flow graph configuratiob.

12.177.3.12 start()

```
void GNSSFlowgraph::start ()
```
Start the flow graph.

12.177.3.13 stop()

```
void GNSSFlowgraph::stop ()
```
Stop the flow graph.

12.177.3.14 stop_duplicated_satellite_channels()

```
void GNSSFlowgraph::stop_duplicated_satellite_channels ()
```
Stops any channel whose decoded satellite is already tracked, on the same signal, by another channel with higher C/N0. The stopped channel's assignment returns to the search pool and the channel is re-dispatched.
GLONASS channels are assigned a frequency channel and only learn their orbital slot from the navigation message, so a false lock can be relabelled as a satellite another channel already tracks; duplicate observations of one satellite break the PVT solution.

12.177.3.15 UpdateVisibilityReference()

```
void GNSSFlowgraph::UpdateVisibilityReference (
            time_t utc_time,
            const std::array< float, 3 > & LLH)
```

Sets the telecommand position/time reference and immediately refreshes visibility. Called by the control thread.

12.177.3.16 visibility_aware_search_enabled()

```
bool GNSSFlowgraph::visibility_aware_search_enabled () const
```

Whether GNSS-SDR.enable_visibility_aware_search is on, so callers can skip the legacy get_visible_sats()/↩ priorize_satellites() startup reorder that MaybeUpdateVisibility() supersedes.

12.177.3.17 wait()

```
void GNSSFlowgraph::wait ()
```

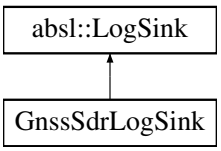
Wait for a flowgraph to complete.
Flowgraphs complete when either (1) all blocks indicate that they are done, or (2) after stop() has been called to request shutdown.
The documentation for this class was generated from the following file:

- gnss_flowgraph.h

12.178 GnssSdrLogSink Class Reference

```
#include <gnss_sdr_log_sink.h>
```

Inheritance diagram for GnssSdrLogSink:



Public Member Functions

- **GnssSdrLogSink** (const std::string &directory, const std::string &basename="gnss-sdr")
- **GnssSdrLogSink** (const GnssSdrLogSink &)=delete
- GnssSdrLogSink & **operator=** (const GnssSdrLogSink &)=delete
- **GnssSdrLogSink** (GnssSdrLogSink &&)=delete
- GnssSdrLogSink & **operator=** (GnssSdrLogSink &&)=delete
- void **Send** (const absl::LogEntry &entry) noexcept override
- void **Flush** () noexcept override
- const std::string & filename () const noexcept

12.178.1 Detailed Description

A sink must be unregistered, and direct callers stopped, before destruction. Directory/file creation errors throw; symlink and runtime I/O errors are reported to stderr. An empty directory selects the system temporary directory.
Definition at line 39 of file gnss_sdr_log_sink.h.

12.178.2 Member Function Documentation

12.178.2.1 filename()

```
const std::string & GnssSdrLogSink::filename () const  [inline], [noexcept]
```

Definition at line 51 of file gnss_sdr_log_sink.h.
The documentation for this class was generated from the following file:

- gnss_sdr_log_sink.h

12.179 GnssSdrLogSinkGuard Class Reference

`#include <gnss_sdr_log_sink.h>`

Public Member Functions

- **GnssSdrLogSinkGuard** (const GnssSdrLogSinkGuard &)=delete
- GnssSdrLogSinkGuard & **operator=** (const GnssSdrLogSinkGuard &)=delete
- **GnssSdrLogSinkGuard** (GnssSdrLogSinkGuard &&)=delete
- GnssSdrLogSinkGuard & **operator=** (GnssSdrLogSinkGuard &&)=delete
- void **Register** (const std::string &directory, const std::string &basename="gnss-sdr")
- void **Shutdown** () noexcept

12.179.1 Detailed Description

Owns a registered sink. Initialize Abseil once before calling Register(). Register/Shutdown belong to the owning thread; Send/Flush may run concurrently.
Definition at line 71 of file gnss_sdr_log_sink.h.
The documentation for this class was generated from the following file:

- gnss_sdr_log_sink.h

12.180 GnssTime Class Reference

Public Attributes

- double rx_time
- int week
- int tow_ms
- double tow_ms_fraction

12.180.1 Detailed Description

Definition at line 23 of file gnss_time.h.

12.180.2 Member Data Documentation

12.180.2.1 rx_time

`double GnssTime::rx_time`
Definition at line 26 of file gnss_time.h.

12.180.2.2 tow_ms

`int GnssTime::tow_ms`
Definition at line 28 of file gnss_time.h.

12.180.2.3 tow_ms_fraction

`double GnssTime::tow_ms_fraction`
Definition at line 29 of file gnss_time.h.

12.180.2.4 week

`int GnssTime::week`
GPS week number (since January 1980)
Definition at line 27 of file gnss_time.h.
The documentation for this class was generated from the following file:

- gnss_time.h

12.181 Gps_Acq_Assist Class Reference

This class is a storage for the GPS GSM RRLL acquisition assistance data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5).
`#include <gps_acq_assist.h>`

Public Member Functions

- Gps_Acq_Assist ()=default

Public Attributes

- uint32_t PRN {}
  SV PRN NUMBER.
- double tow {}
  Time Of Week assigned to the acquisition data.
- double Doppler0 {}
  Doppler (0 order term) [Hz].
- double Doppler1 {}
  Doppler (1 order term) [Hz].
- double dopplerUncertainty {}
  Doppler Uncertainty [Hz].
- double Code_Phase {}
  Code phase [chips].
- double Code_Phase_int {}
  Integer Code Phase [1 C/A code period].
- double GPS_Bit_Number {}
  GPS Bit Number.
- double Code_Phase_window {}
  Code Phase search window [chips].
- double Azimuth {}
  Satellite Azimuth [deg].
- double Elevation {}
  Satellite Elevation [deg].

12.181.1 Detailed Description

This class is a storage for the GPS GSM RRLL acquisition assistance data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5).
Definition at line 37 of file gps_acq_assist.h.

12.181.2 Constructor & Destructor Documentation

12.181.2.1 Gps_Acq_Assist()

`Gps_Acq_Assist::Gps_Acq_Assist () [default]`
Default constructor

12.181.3 Member Data Documentation

12.181.3.1 Azimuth

`double Gps_Acq_Assist::Azimuth {}`
Satellite Azimuth [deg].
Definition at line 54 of file gps_acq_assist.h.

12.181.3.2 Code_Phase

```
double Gps_Acq_Assist::Code_Phase {}
```
Code phase [chips].
Definition at line 50 of file gps_acq_assist.h.

12.181.3.3 Code_Phase_int

```
double Gps_Acq_Assist::Code_Phase_int {}
```
Integer Code Phase [1 C/A code period].
Definition at line 51 of file gps_acq_assist.h.

12.181.3.4 Code_Phase_window

```
double Gps_Acq_Assist::Code_Phase_window {}
```
Code Phase search window [chips].
Definition at line 53 of file gps_acq_assist.h.

12.181.3.5 Doppler0

```
double Gps_Acq_Assist::Doppler0 {}
```
Doppler (0 order term) [Hz].
Definition at line 47 of file gps_acq_assist.h.

12.181.3.6 Doppler1

```
double Gps_Acq_Assist::Doppler1 {}
```
Doppler (1 order term) [Hz].
Definition at line 48 of file gps_acq_assist.h.

12.181.3.7 dopplerUncertainty

```
double Gps_Acq_Assist::dopplerUncertainty {}
```
Doppler Uncertainty [Hz].
Definition at line 49 of file gps_acq_assist.h.

12.181.3.8 Elevation

```
double Gps_Acq_Assist::Elevation {}
```
Satellite Elevation [deg].
Definition at line 55 of file gps_acq_assist.h.

12.181.3.9 GPS_Bit_Number

```
double Gps_Acq_Assist::GPS_Bit_Number {}
```
GPS Bit Number.
Definition at line 52 of file gps_acq_assist.h.

12.181.3.10 PRN

```
uint32_t Gps_Acq_Assist::PRN {}
```
SV PRN NUMBER.
Definition at line 45 of file gps_acq_assist.h.

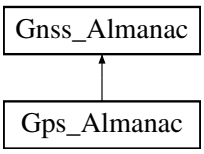
12.181.3.11 tow

```
double Gps_Acq_Assist::tow {}
```
Time Of Week assigned to the acquisition data.
Definition at line 46 of file gps_acq_assist.h.
The documentation for this class was generated from the following file:

- gps_acq_assist.h

12.182 Gps_Almanac Class Reference

This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M.
`#include <gps_almanac.h>`
Inheritance diagram for Gps_Almanac:



Public Member Functions

- Gps_Almanac ()
- void set_system (char system)
- char get_system () const
- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

Public Member Functions inherited from Gnss_Almanac

- Gnss_Almanac ()=default
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosVelComputation (double transmitTime, std::array< double, 7 > &pos_vel_dtr) const

  Computes satellite Position and Velocity, in ECEF, for a given time (expressed in seconds of week).

Public Attributes

- int32_t SV_health {}

  SV Health.
- int32_t AS_status {}

  Anti-Spoofing Flags and SV Configuration.

Public Attributes inherited from Gnss_Almanac

- uint32_t PRN {}

  SV PRN NUMBER.
- double delta_i {}

  Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles).
- int32_t toa {}

  Almanac data reference time of week [s].
- int32_t WNa {}

  Almanac week number.
- double M_0 {}

  Mean Anomaly at Reference Time [semi-circles].
- double ecc {}

  Eccentricity [dimensionless].
- double sqrtA {}

  Square Root of the Semi-Major Axis [sqrt(m)].

- double OMEGA_0 {}

  Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles].
- double omega {}

  Argument of Perigee [semi-cicles].
- double OMEGAdot {}

  Rate of Right Ascension [semi-circles/s].
- double af0 {}

  Coefficient 0 of code phase offset model [s].
- double af1 {}

  Coefficient 1 of code phase offset model [s/s].
- bool from_startup_load {false}

  Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the broadcast. Runtime-only provenance flag, intentionally not serialized.

Additional Inherited Members

Protected Attributes inherited from Gnss_Almanac

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J': QZSS.

12.182.1 Detailed Description

This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
Definition at line 35 of file gps_almanac.h.

12.182.2 Constructor & Destructor Documentation

12.182.2.1 Gps_Almanac()

```
Gps_Almanac::Gps_Almanac ()  [inline]
```

Default constructor
Definition at line 41 of file gps_almanac.h.
References Gnss_Almanac::System.

12.182.3 Member Function Documentation

12.182.3.1 get_system()

```
char Gps_Almanac::get_system () const  [inline]
```

Definition at line 51 of file gps_almanac.h.

12.182.3.2 serialize()

```
template<class Archive>
void Gps_Almanac::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

Definition at line 61 of file gps_almanac.h.

12.182.3.3 set_system()

```
void Gps_Almanac::set_system (
            char system)  [inline]
```

Definition at line 46 of file gps_almanac.h.

12.182.4 Member Data Documentation

12.182.4.1 AS_status

`int32_t Gps_Almanac::AS_status {}`
Anti-Spoofing Flags and SV Configuration.
Definition at line 57 of file gps_almanac.h.

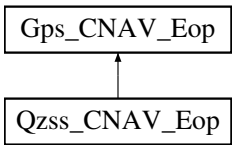
12.182.4.2 SV_health

`int32_t Gps_Almanac::SV_health {}`
SV Health.
Definition at line 56 of file gps_almanac.h.
The documentation for this class was generated from the following file:

- gps_almanac.h

12.183 Gps_CNAV_Eop Class Reference

Storage for the Earth orientation parameters broadcast in GPS/QZSS CNAV Message Type 32, as described in IS-GPS-200M.
`#include <gps_cnav_eop.h>`
Inheritance diagram for Gps_CNAV_Eop:



Public Attributes

- uint32_t PRN {}
  Transmitting satellite PRN.
- double tow {}
  Message transmission time [s of GPS week].
- double t_eop {}
  EOP reference time [s of GPS week].
- double pm_x {}
  X-axis polar motion [arc-seconds].
- double pm_x_dot {}
  X-axis polar motion rate [arc-seconds/day].
- double pm_y {}
  Y-axis polar motion [arc-seconds].
- double pm_y_dot {}
  Y-axis polar motion rate [arc-seconds/day].
- double delta_ut1_gps {}
  UT1-GPS difference [s].
- double delta_ut1_gps_dot {}
  UT1-GPS difference rate [s/day].
- bool valid {}

12.183.1 Detailed Description

Storage for the Earth orientation parameters broadcast in GPS/QZSS CNAV Message Type 32, as described in IS-GPS-200M.
Definition at line 34 of file gps_cnav_eop.h.

12.183.2 Member Data Documentation

12.183.2.1 delta_ut1_gps

```
double Gps_CNAV_Eop::delta_ut1_gps {}
```

UT1-GPS difference [s].
Definition at line 44 of file gps_cnav_eop.h.

12.183.2.2 delta_ut1_gps_dot

```
double Gps_CNAV_Eop::delta_ut1_gps_dot {}
```

UT1-GPS difference rate [s/day].
Definition at line 45 of file gps_cnav_eop.h.

12.183.2.3 pm_x

```
double Gps_CNAV_Eop::pm_x {}
```

X-axis polar motion [arc-seconds].
Definition at line 40 of file gps_cnav_eop.h.

12.183.2.4 pm_x_dot

```
double Gps_CNAV_Eop::pm_x_dot {}
```

X-axis polar motion rate [arc-seconds/day].
Definition at line 41 of file gps_cnav_eop.h.

12.183.2.5 pm_y

```
double Gps_CNAV_Eop::pm_y {}
```

Y-axis polar motion [arc-seconds].
Definition at line 42 of file gps_cnav_eop.h.

12.183.2.6 pm_y_dot

```
double Gps_CNAV_Eop::pm_y_dot {}
```

Y-axis polar motion rate [arc-seconds/day].
Definition at line 43 of file gps_cnav_eop.h.

12.183.2.7 PRN

```
uint32_t Gps_CNAV_Eop::PRN {}
```

Transmitting satellite PRN.
Definition at line 37 of file gps_cnav_eop.h.

12.183.2.8 t_eop

```
double Gps_CNAV_Eop::t_eop {}
```

EOP reference time [s of GPS week].
Definition at line 39 of file gps_cnav_eop.h.

12.183.2.9 tow

```
double Gps_CNAV_Eop::tow {}
```

Message transmission time [s of GPS week].
Definition at line 38 of file gps_cnav_eop.h.

12.183.2.10 valid

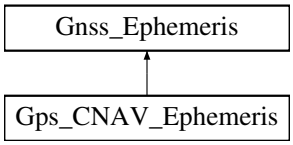
```
bool Gps_CNAV_Eop::valid {}
```

Definition at line 46 of file gps_cnav_eop.h.
The documentation for this class was generated from the following file:

- gps_cnav_eop.h

12.184 Gps_CNAV_Ephemeris Class Reference

This is a storage class for the GPS CNAV ephemeris data as described in IS-GPS-200M.
`#include <gps_cnav_ephemeris.h>`
Inheritance diagram for Gps_CNAV_Ephemeris:



Public Member Functions

- Gps_CNAV_Ephemeris ()
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Member Functions inherited from Gnss_Ephemeris

- double sv_clock_drift (double transmitTime)

  Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- double delta_A {}

  Semi-major axis difference at reference time.
- double delta_OMEGAdot {}

  Rate of Right Ascension difference [semi-circles/s].
- int32_t toe1 {}

  Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M) [s].
- int32_t toe2 {}

  Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M) [s].
- int32_t signal_health {}

  Signal health (L1/L2/L5).
- int32_t WNop {-1}

  CEI data sequence propagation week number from MT30, modulo 256.
- int32_t top {}

  Data predict time of week.
- int32_t URAED {}

  ED Accuracy Index from MT10.
- int32_t URANED0 {}

  NED Accuracy Index from MT30-37.

- int32_t URANED1 {}

  NED Accuracy Change Index from MT30-37.
- int32_t URANED2 {}

  NED Accuracy Change Rate Index from MT30-37.
- int32_t URA {}

  Legacy alias for URAED.
- double URA0 {}

  Legacy alias for URANED0.
- double URA1 {}

  Legacy alias for URANED1.
- double URA2 {}

  Legacy alias for URANED2.
- double TGD {}

  Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s].
- double ISCL1 {}
- double ISCL2 {}
- double ISCL5I {}
- double ISCL5Q {}
- bool integrity_status_flag {}

  If true, enhanced level of integrity assurance.
- bool l2c_phasing_flag {}
- bool alert_flag {}

  If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
- bool antispoofing_flag {}

  If true, the AntiSpoofing mode is ON in that SV.

Public Attributes inherited from Gnss_Ephemeris

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

  Square root of the semi-major axis [meters^ 1/2].
- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
- double OMEGA_0 {}

  Longitude of ascending node of orbital plane at weekly epoch [rad].
- double i_0 {}

  Inclination angle at reference time [rad].
- double omega {}

  Argument of perigee [rad].
- double OMEGAdot {}

Rate of right ascension [rad/sec].

- double idot {}

  Rate of inclination angle [rad/sec].

- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].

- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].

- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].

- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].

- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].

- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].

- int32_t toe {}

  Ephemeris reference time [s].

- int32_t toc {}

  Clock correction data reference Time of Week [sec].

- double af0 {}

  SV clock bias correction coefficient [s].

- double af1 {}

  SV clock drift correction coefficient [s/s].

- double af2 {}

  SV clock drift rate correction coefficient [s/s^ 2].

- double satClkDrift {}

  SV clock drift.

- double dtr {}

  Relativistic clock correction term.

- int32_t WN {}

  Week number.

- int32_t tow {}

  Time of Week.

- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.

- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

- double satpos_Z {}

  Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).

- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].

- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].

- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Additional Inherited Members

Protected Attributes inherited from Gnss_Ephemeris

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.184.1 Detailed Description

This is a storage class for the GPS CNAV ephemeris data as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
Definition at line 38 of file gps_cnav_ephemeris.h.

12.184.2 Constructor & Destructor Documentation

12.184.2.1 Gps_CNAV_Ephemeris()

```
Gps_CNAV_Ephemeris::Gps_CNAV_Ephemeris ()  [inline]
```

Constructor
Definition at line 44 of file gps_cnav_ephemeris.h.
References Gnss_Ephemeris::System.

12.184.3 Member Function Documentation

12.184.3.1 serialize()

```
template<class Archive>
void Gps_CNAV_Ephemeris::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.
Definition at line 98 of file gps_cnav_ephemeris.h.
References Gnss_Ephemeris::Adot, Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, alert_flag, antispoofing_flag, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, delta_A, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::delta_ndot, delta_OMEGAdot, Gnss_Ephemeris::dtr, Gnss_Ephemeris::ecc, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, integrity_status_flag, Gnss_Ephemeris::M_0, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdo Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, Gnss_Ephemeris::sqrtA, TGD, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, toe1, toe2, top, Gnss_Ephemeris::tow, URA, URA0, URA1, URA2, URAED, URANED0, URANED1, URANED2, Gnss_Ephemeris::WN, and WNop.

12.184.4 Member Data Documentation

12.184.4.1 alert_flag

```
bool Gps_CNAV_Ephemeris::alert_flag {}
```

If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
Definition at line 89 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.2 antispoofing_flag

```
bool Gps_CNAV_Ephemeris::antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.
Definition at line 90 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.3 delta_A

```
double Gps_CNAV_Ephemeris::delta_A {}
```
Semi-major axis difference at reference time.
Definition at line 49 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.4 delta_OMEGAdot

```
double Gps_CNAV_Ephemeris::delta_OMEGAdot {}
```
Rate of Right Ascension difference [semi-circles/s].
Definition at line 50 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.5 integrity_status_flag

```
bool Gps_CNAV_Ephemeris::integrity_status_flag {}
```
If true, enhanced level of integrity assurance.
If false, indicates that the conveying signal is provided with the legacy level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 4.42 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-5 per hour.
If true, indicates that the conveying signal is provided with an enhanced level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 5.73 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-8 per hour.
Definition at line 87 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.6 ISCL1

```
double Gps_CNAV_Ephemeris::ISCL1 {}
```
Definition at line 69 of file gps_cnav_ephemeris.h.

12.184.4.7 ISCL2

```
double Gps_CNAV_Ephemeris::ISCL2 {}
```
Definition at line 70 of file gps_cnav_ephemeris.h.

12.184.4.8 ISCL5I

```
double Gps_CNAV_Ephemeris::ISCL5I {}
```
Definition at line 71 of file gps_cnav_ephemeris.h.

12.184.4.9 ISCL5Q

```
double Gps_CNAV_Ephemeris::ISCL5Q {}
```
Definition at line 72 of file gps_cnav_ephemeris.h.

12.184.4.10 l2c_phasing_flag

```
bool Gps_CNAV_Ephemeris::l2c_phasing_flag {}
```
Definition at line 88 of file gps_cnav_ephemeris.h.

12.184.4.11 signal_health

```
int32_t Gps_CNAV_Ephemeris::signal_health {}
```
Signal health (L1/L2/L5).
Definition at line 53 of file gps_cnav_ephemeris.h.

12.184.4.12 TGD

```
double Gps_CNAV_Ephemeris::TGD {}
```
Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s].
Definition at line 68 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.13 toe1

```
int32_t Gps_CNAV_Ephemeris::toe1 {}
```
Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M) [s].
Definition at line 51 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.14 toe2

```
int32_t Gps_CNAV_Ephemeris::toe2 {}
```
Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M) [s].
Definition at line 52 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.15 top

```
int32_t Gps_CNAV_Ephemeris::top {}
```
Data predict time of week.
Definition at line 55 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.16 URA

```
int32_t Gps_CNAV_Ephemeris::URA {}
```
Legacy alias for URAED.
Definition at line 62 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.17 URA0

```
double Gps_CNAV_Ephemeris::URA0 {}
```
Legacy alias for URANED0.
Definition at line 63 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.18 URA1

```
double Gps_CNAV_Ephemeris::URA1 {}
```
Legacy alias for URANED1.
Definition at line 64 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.19 URA2

```
double Gps_CNAV_Ephemeris::URA2 {}
```
Legacy alias for URANED2.
Definition at line 65 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.20 URAED

```
int32_t Gps_CNAV_Ephemeris::URAED {}
```
ED Accuracy Index from MT10.

Definition at line 56 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.21 URANED0

`int32_t Gps_CNAV_Ephemeris::URANED0 {}`
NED Accuracy Index from MT30-37.
Definition at line 58 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.22 URANED1

`int32_t Gps_CNAV_Ephemeris::URANED1 {}`
NED Accuracy Change Index from MT30-37.
Definition at line 59 of file gps_cnav_ephemeris.h.
Referenced by serialize().

12.184.4.23 URANED2

`int32_t Gps_CNAV_Ephemeris::URANED2 {}`
NED Accuracy Change Rate Index from MT30-37.
Definition at line 60 of file gps_cnav_ephemeris.h.
Referenced by serialize().

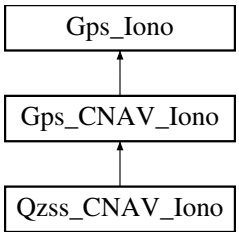
12.184.4.24 WNop

`int32_t Gps_CNAV_Ephemeris::WNop {-1}`
CEI data sequence propagation week number from MT30, modulo 256.
Definition at line 54 of file gps_cnav_ephemeris.h.
Referenced by serialize().
The documentation for this class was generated from the following file:

- gps_cnav_ephemeris.h

12.185 Gps_CNAV_Iono Class Reference

This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
`#include <gps_cnav_iono.h>`
Inheritance diagram for Gps_CNAV_Iono:



Public Member Functions

- Gps_CNAV_Iono ()=default

  Default constructor.

Public Member Functions inherited from Gps_Iono

- Gps_Iono ()=default

  Default constructor.

- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Additional Inherited Members

Public Attributes inherited from Gps_Iono

- double alpha0 {}

  Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
- double alpha1 {}

  Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].
- double alpha2 {}

  Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].
- double alpha3 {}

  Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].
- double beta0 {}

  Coefficient 0 of a cubic equation representing the period of the model [s].
- double beta1 {}

  Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].
- double beta2 {}

  Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].
- double beta3 {}

  Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].
- bool valid {}

  Valid flag.

12.185.1 Detailed Description

This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
Definition at line 34 of file gps_cnav_iono.h.

12.185.2 Constructor & Destructor Documentation

12.185.2.1 Gps_CNAV_Iono()

```
Gps_CNAV_Iono::Gps_CNAV_Iono ()  [default]
```

Default constructor.
Referenced by Qzss_CNAV_Iono::Qzss_CNAV_Iono().
The documentation for this class was generated from the following file:

- gps_cnav_iono.h

12.186 Gps_CNAV_Navigation_Message Class Reference

This class decodes a GPS CNAV Data message as described in IS-GPS-200M.
`#include <gps_cnav_navigation_message.h>`

Public Member Functions

- Gps_CNAV_Navigation_Message (CnavSystem system=CnavSystem::GPS)
- void **decode_page** (const std::bitset< GPS_CNAV_DATA_PAGE_BITS > &data_bits)
- Gps_CNAV_Ephemeris get_ephemeris () const

  Obtain a GPS SV Ephemeris class filled with current SV data.
- bool have_new_eop ()

  Check if a newly decoded CNAV Earth orientation record is available.
- Gps_CNAV_Eop get_eop () const

  Obtain the latest CNAV Earth orientation parameters.
- bool have_new_iono ()

  Check if we have a new iono record stored in the GPS ephemeris class.
- Gps_CNAV_Iono get_iono () const

  Obtain a GPS ionospheric correction parameters class filled with current SV data.
- Gps_CNAV_Utc_Model get_utc_model ()

  Obtain a GPS UTC model parameters class filled with current SV data.
- bool have_new_utc_model ()
- bool have_new_ephemeris ()

  Check if we have a new ephemeris stored in the GPS ephemeris class.

12.186.1 Detailed Description

This class decodes a GPS CNAV Data message as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
Definition at line 51 of file gps_cnav_navigation_message.h.

12.186.2 Constructor & Destructor Documentation

12.186.2.1 Gps_CNAV_Navigation_Message()

```
Gps_CNAV_Navigation_Message::Gps_CNAV_Navigation_Message (
            CnavSystem system = CnavSystem::GPS)  [explicit]
```

Default constructor

12.186.3 Member Function Documentation

12.186.3.1 get_eop()

```
Gps_CNAV_Eop Gps_CNAV_Navigation_Message::get_eop () const
```

Obtain the latest CNAV Earth orientation parameters.

12.186.3.2 get_ephemeris()

```
Gps_CNAV_Ephemeris Gps_CNAV_Navigation_Message::get_ephemeris () const
```

Obtain a GPS SV Ephemeris class filled with current SV data.

12.186.3.3 get_iono()

```
Gps_CNAV_Iono Gps_CNAV_Navigation_Message::get_iono () const
```

Obtain a GPS ionospheric correction parameters class filled with current SV data.

12.186.3.4 get_utc_model()

```
Gps_CNAV_Utc_Model Gps_CNAV_Navigation_Message::get_utc_model ()
```

Obtain a GPS UTC model parameters class filled with current SV data.

12.186.3.5 have_new_eop()

`bool Gps_CNAV_Navigation_Message::have_new_eop ()`
Check if a newly decoded CNAV Earth orientation record is available.

12.186.3.6 have_new_ephemeris()

`bool Gps_CNAV_Navigation_Message::have_new_ephemeris ()`
Check if we have a new ephemeris stored in the GPS ephemeris class.

12.186.3.7 have_new_iono()

`bool Gps_CNAV_Navigation_Message::have_new_iono ()`
Check if we have a new iono record stored in the GPS ephemeris class.

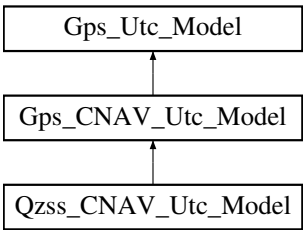
12.186.3.8 have_new_utc_model()

`bool Gps_CNAV_Navigation_Message::have_new_utc_model ()`
\briefCheck if we have a new GPS UTC model record stored in the GPS ephemeris class
The documentation for this class was generated from the following file:

- gps_cnav_navigation_message.h

12.187 Gps_CNAV_Utc_Model Class Reference

This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M.
`#include <gps_cnav_utc_model.h>`
Inheritance diagram for Gps_CNAV_Utc_Model:



Public Member Functions

- Gps_CNAV_Utc_Model ()=default
  Default constructor.

Public Member Functions inherited from Gps_Utc_Model

- Gps_Utc_Model ()=default
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

Additional Inherited Members

Public Attributes inherited from Gps_Utc_Model

- double A0 {}
  Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s].
- double A1 {}
  1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- double A2 {}

2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]

- int32_t tot {}

  Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200M) [s].
- int32_t WN_T {}

  UTC reference week number [weeks].
- int32_t DeltaT_LS {}

  Delta time due to leap seconds [s]. Number of leap seconds since 6-Jan-1980 as transmitted by the GPS almanac.
- int32_t WN_LSF {}

  Week number at the end of which the leap second becomes effective [weeks].
- int32_t DN {}

  Day number (DN) at the end of which the leap second becomes effective [days].
- int32_t DeltaT_LSF {}

  Scheduled future or recent past (relative to NAV message upload) value of the delta time due to leap seconds [s].
- bool valid {}

12.187.1 Detailed Description

This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
Definition at line 34 of file gps_cnav_utc_model.h.

12.187.2 Constructor & Destructor Documentation

12.187.2.1 Gps_CNAV_Utc_Model()

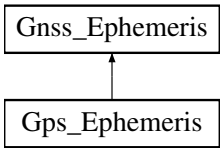
```
Gps_CNAV_Utc_Model::Gps_CNAV_Utc_Model () [default]
```

Default constructor.
Referenced by Qzss_CNAV_Utc_Model::Qzss_CNAV_Utc_Model().
The documentation for this class was generated from the following file:

- gps_cnav_utc_model.h

12.188 Gps_Ephemeris Class Reference

This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200M.
`#include <gps_ephemeris.h>`
Inheritance diagram for Gps_Ephemeris:



Public Member Functions

- Gps_Ephemeris ()
- void set_system (char system)
- char get_system () const
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Member Functions inherited from Gnss_Ephemeris

- double sv_clock_drift (double transmitTime)

  Sets (satClkDrift) and (dtr), and returns the clock drift in seconds according to the User Algorithm for SV Clock Correction (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
- double predicted_doppler (double rx_time_s, double lat, double lon, double h, double ve, double vn, double vu, int band) const

  Computes prediction of the Doppler shift for a given time and receiver's position and velocity.
- void satellitePosition (double transmitTime)

  Computes the ECEF SV coordinates and ECEF velocity.

Public Attributes

- int32_t code_on_L2 {}

  If 1, P code ON in L2; if 2, C/A code ON in L2;.
- bool L2_P_data_flag {}

  When true, indicates that the NAV data stream was commanded OFF on the P-code of the L2 channel.
- int32_t SV_accuracy {}

  User Range Accuracy (URA) index of the SV (reference paragraph 6.2.1) for the standard positioning service user (Ref 20.3.3.3.1.3 IS-GPS-200M).
- int32_t SV_health {}

  Satellite heath status.
- double TGD {}

  Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s].
- int32_t IODC {}

  Issue of Data, Clock.
- int32_t IODE_SF2 {}

  Issue of Data, Ephemeris (IODE), subframe 2.
- int32_t IODE_SF3 {}

  Issue of Data, Ephemeris (IODE), subframe 3.
- int32_t AODO {}

  Age of Data Offset (AODO) term for the navigation message correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s].
- bool fit_interval_flag {}

  indicates the curve-fit interval used by the CS (Block II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4 hours, 1 = greater than 4 hours.
- double spare1 {}
- double spare2 {}
- bool integrity_status_flag {}

  If true, enhanced level of integrity assurance.
- bool alert_flag {}

  If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
- bool antispoofing_flag {}

  If true, the AntiSpoofing mode is ON in that SV.
- std::map< int, std::string > satelliteBlock

  Map that stores to which block the PRN belongs https://www.navcen.uscg.gov/?Do=constellationStatus.

Public Attributes inherited from Gnss_Ephemeris

- uint32_t PRN {}

  SV ID.
- double M_0 {}

  Mean anomaly at reference time [rad].
- double Adot {}

  Change rate in semi-major axis (CNAV).
- double delta_ndot {}

  Rate of mean motion difference from computed value (CNAV).
- double delta_n {}

  Mean motion difference from computed value [rad/sec].
- double ecc {}

  Eccentricity.
- double sqrtA {}

  Square root of the semi-major axis [meters^ 1/2].
- double A0 {}

  Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set.
- double OMEGA_0 {}

  Longitude of ascending node of orbital plane at weekly epoch [rad].
- double i_0 {}

  Inclination angle at reference time [rad].
- double omega {}

  Argument of perigee [rad].
- double OMEGAdot {}

  Rate of right ascension [rad/sec].
- double idot {}

  Rate of inclination angle [rad/sec].
- double Cuc {}

  Amplitude of the cosine harmonic correction term to the argument of latitude [rad].
- double Cus {}

  Amplitude of the sine harmonic correction term to the argument of latitude [rad].
- double Crc {}

  Amplitude of the cosine harmonic correction term to the orbit radius [meters].
- double Crs {}

  Amplitude of the sine harmonic correction term to the orbit radius [meters].
- double Cic {}

  Amplitude of the cosine harmonic correction term to the angle of inclination [rad].
- double Cis {}

  Amplitude of the sine harmonic correction term to the angle of inclination [rad].
- int32_t toe {}

  Ephemeris reference time [s].
- int32_t toc {}

  Clock correction data reference Time of Week [sec].
- double af0 {}

  SV clock bias correction coefficient [s].
- double af1 {}

  SV clock drift correction coefficient [s/s].
- double af2 {}

  SV clock drift rate correction coefficient [s/s^ 2].
- double satClkDrift {}

SV clock drift.

- double dtr {}

  Relativistic clock correction term.

- int32_t WN {}

  Week number.

- int32_t tow {}

  Time of Week.

- double satpos_X {}

  Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.

- double satpos_Y {}

  Earth-fixed coordinate y of the satellite [m]. Completes a right-handed, Earth-Centered, Earth-Fixed orthogonal coordinate system.

- double satpos_Z {}

  Earth-fixed coordinate z of the satellite [m]. The direction of the IERS (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).

- double satvel_X {}

  Earth-fixed velocity coordinate x of the satellite [m].

- double satvel_Y {}

  Earth-fixed velocity coordinate y of the satellite [m].

- double satvel_Z {}

  Earth-fixed velocity coordinate z of the satellite [m].

Additional Inherited Members

Protected Attributes inherited from Gnss_Ephemeris

- char System {}

  Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou.

12.188.1 Detailed Description

This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200M. See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
Definition at line 40 of file gps_ephemeris.h.

12.188.2 Constructor & Destructor Documentation

12.188.2.1 Gps_Ephemeris()

```
Gps_Ephemeris::Gps_Ephemeris ()
```

Default constructor
References Gnss_Ephemeris::System.

12.188.3 Member Function Documentation

12.188.3.1 get_system()

```
char Gps_Ephemeris::get_system () const  [inline]
```

Definition at line 53 of file gps_ephemeris.h.

12.188.3.2 serialize()

```
template<class Archive>
void Gps_Ephemeris::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.
Definition at line 96 of file gps_ephemeris.h.
References Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, alert_flag, antispoofing_flag, AODO, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, code_on_L2, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::dtr, Gnss_Ephemeris::ecc, fit_interval_flag, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, integrity_status_flag, IODC, IODE_SF2, IODE_SF3, L2_P_data_flag, Gnss_Ephemeris::M_0, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdot, Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, Gnss_Ephemeris::sqrtA, SV_accuracy, SV_health, TGD, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, Gnss_Ephemeris::tow, and Gnss_Ephemeris::WN.

12.188.3.3 set_system()

```
void Gps_Ephemeris::set_system (
            char system)  [inline]
```

Definition at line 48 of file gps_ephemeris.h.

12.188.4 Member Data Documentation

12.188.4.1 alert_flag

```
bool Gps_Ephemeris::alert_flag {}
```

If true, indicates that the SV URA may be worse than indicated in d_SV_accuracy, use that SV at our own risk.
Definition at line 85 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.2 antispoofing_flag

```
bool Gps_Ephemeris::antispoofing_flag {}
```

If true, the AntiSpoofing mode is ON in that SV.
Definition at line 86 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.3 AODO

```
int32_t Gps_Ephemeris::AODO {}
```

Age of Data Offset (AODO) term for the navigation message correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s].
Definition at line 66 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.4 code_on_L2

```
int32_t Gps_Ephemeris::code_on_L2 {}
```

If 1, P code ON in L2; if 2, C/A code ON in L2;.
Definition at line 58 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.5 fit_interval_flag

```
bool Gps_Ephemeris::fit_interval_flag {}
```

indicates the curve-fit interval used by the CS (Block II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4 hours, 1 = greater than 4 hours.

Definition at line 68 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.6 integrity_status_flag

```
bool Gps_Ephemeris::integrity_status_flag {}
```

If true, enhanced level of integrity assurance.
If false, indicates that the conveying signal is provided with the legacy level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 4.42 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-5 per hour.
If true, indicates that the conveying signal is provided with an enhanced level of integrity assurance. That is, the probability that the instantaneous URE of the conveying signal exceeds 5.73 times the upper bound value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying alert, is less than 1E-8 per hour.
Definition at line 84 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.7 IODC

```
int32_t Gps_Ephemeris::IODC {}
```

Issue of Data, Clock.
Definition at line 63 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.8 IODE_SF2

```
int32_t Gps_Ephemeris::IODE_SF2 {}
```

Issue of Data, Ephemeris (IODE), subframe 2.
Definition at line 64 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.9 IODE_SF3

```
int32_t Gps_Ephemeris::IODE_SF3 {}
```

Issue of Data, Ephemeris (IODE), subframe 3.
Definition at line 65 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.10 L2_P_data_flag

```
bool Gps_Ephemeris::L2_P_data_flag {}
```

When true, indicates that the NAV data stream was commanded OFF on the P-code of the L2 channel.
Definition at line 59 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.11 satelliteBlock

```
std::map<int, std::string> Gps_Ephemeris::satelliteBlock
```

Map that stores to which block the PRN belongs https://www.navcen.uscg.gov/?Do=constellation↩Status.
Definition at line 88 of file gps_ephemeris.h.

12.188.4.12 spare1

```
double Gps_Ephemeris::spare1 {}
```

Definition at line 69 of file gps_ephemeris.h.

12.188.4.13 spare2

```
double Gps_Ephemeris::spare2 {}
```

Definition at line 70 of file gps_ephemeris.h.

12.188.4.14 SV_accuracy

`int32_t Gps_Ephemeris::SV_accuracy {}`
User Range Accuracy (URA) index of the SV (reference paragraph 6.2.1) for the standard positioning service user (Ref 20.3.3.3.1.3 IS-GPS-200M).
Definition at line 60 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

12.188.4.15 SV_health

`int32_t Gps_Ephemeris::SV_health {}`
Satellite heath status.
Definition at line 61 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().

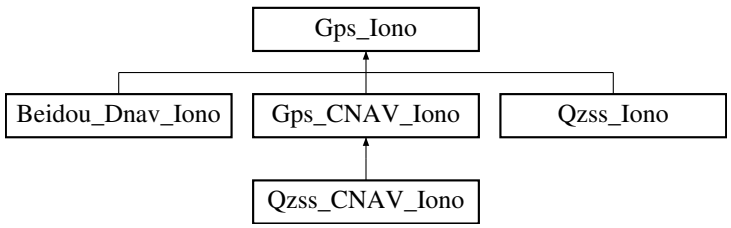
12.188.4.16 TGD

`double Gps_Ephemeris::TGD {}`
Estimated Group Delay Differential: L1-L2 correction term only for the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s].
Definition at line 62 of file gps_ephemeris.h.
Referenced by Serdes_Gps_Eph::readProtobuffer(), and serialize().
The documentation for this class was generated from the following file:

- gps_ephemeris.h

12.189 Gps_Iono Class Reference

This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
`#include <gps_iono.h>`
Inheritance diagram for Gps_Iono:



Public Member Functions

- Gps_Iono ()=default

  Default constructor.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Public Attributes

- double alpha0 {}

  Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
- double alpha1 {}

  Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].
- double alpha2 {}

  Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].

- double alpha3 {}

  Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].
- double beta0 {}

  Coefficient 0 of a cubic equation representing the period of the model [s].
- double beta1 {}

  Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].
- double beta2 {}

  Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].
- double beta3 {}

  Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].
- bool valid {}

  Valid flag.

12.189.1 Detailed Description

This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
Definition at line 35 of file gps_iono.h.

12.189.2 Constructor & Destructor Documentation

12.189.2.1 Gps_Iono()

```
Gps_Iono::Gps_Iono ()  [default]
```

Default constructor.
Referenced by Qzss_Iono::Qzss_Iono().

12.189.3 Member Function Documentation

12.189.3.1 serialize()

```
template<class Archive>
void Gps_Iono::serialize (
            Archive & archive,
            const unsigned int version)  [inline]
```

Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.
Definition at line 58 of file gps_iono.h.
References alpha0, alpha1, alpha2, alpha3, beta0, beta1, beta2, and beta3.

12.189.4 Member Data Documentation

12.189.4.1 alpha0

```
double Gps_Iono::alpha0 {}
```

Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
Definition at line 41 of file gps_iono.h.
Referenced by serialize().

12.189.4.2 alpha1

```
double Gps_Iono::alpha1 {}
```

Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].
Definition at line 42 of file gps_iono.h.
Referenced by serialize().

12.189.4.3 alpha2

`double Gps_Iono::alpha2 {}`
Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].
Definition at line 43 of file gps_iono.h.
Referenced by serialize().

12.189.4.4 alpha3

`double Gps_Iono::alpha3 {}`
Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].
Definition at line 44 of file gps_iono.h.
Referenced by serialize().

12.189.4.5 beta0

`double Gps_Iono::beta0 {}`
Coefficient 0 of a cubic equation representing the period of the model [s].
Definition at line 45 of file gps_iono.h.
Referenced by serialize().

12.189.4.6 beta1

`double Gps_Iono::beta1 {}`
Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].
Definition at line 46 of file gps_iono.h.
Referenced by serialize().

12.189.4.7 beta2

`double Gps_Iono::beta2 {}`
Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].
Definition at line 47 of file gps_iono.h.
Referenced by serialize().

12.189.4.8 beta3

`double Gps_Iono::beta3 {}`
Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].
Definition at line 48 of file gps_iono.h.
Referenced by serialize().

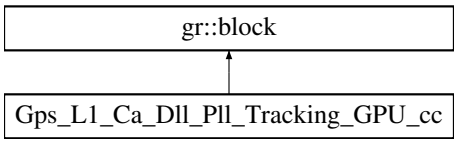
12.189.4.9 valid

`bool Gps_Iono::valid {}`
Valid flag.
Definition at line 50 of file gps_iono.h.
The documentation for this class was generated from the following file:

- gps_iono.h

12.190 Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc Class Reference

This class implements a DLL + PLL tracking loop block.
`#include <gps_l1_ca_dll_pll_tracking_gpu_cc.h>`
Inheritance diagram for Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro *p_gnss_synchro)
- void **start_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)

Friends

- gps_l1_ca_dll_pll_tracking_gpu_cc_sptr **gps_l1_ca_dll_pll_make_tracking_gpu_cc** (int64_t fs_in, uint32_t vector_length, bool dump, std::string dump_filename, float pll_bw_hz, float dll_bw_hz, float early_↩ late_space_chips)

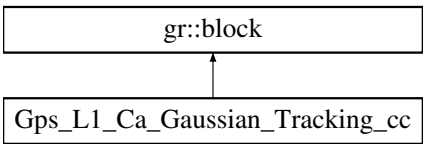
12.190.1 Detailed Description

This class implements a DLL + PLL tracking loop block.
Definition at line 60 of file gps_l1_ca_dll_pll_tracking_gpu_cc.h.
The documentation for this class was generated from the following file:

- gps_l1_ca_dll_pll_tracking_gpu_cc.h

12.191 Gps_L1_Ca_Gaussian_Tracking_cc Class Reference

This class implements a DLL + PLL tracking loop block.
`#include <gps_l1_ca_gaussian_tracking_cc.h>`
Inheritance diagram for Gps_L1_Ca_Gaussian_Tracking_cc:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro *p_gnss_synchro)
- void **start_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)

Friends

- gps_l1_ca_gaussian_tracking_cc_sptr **gps_l1_ca_gaussian_make_tracking_cc** (uint32_t order, int64_↩ t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float dll_bw_hz, float early↩ _late_space_chips, bool bce_run, uint32_t bce_ptrans, uint32_t bce_strans, int32_t bce_nu, int32_t bce_↩ kappa)

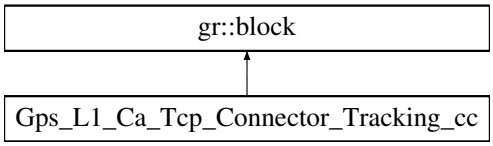
12.191.1 Detailed Description

This class implements a DLL + PLL tracking loop block.
Definition at line 74 of file gps_l1_ca_gaussian_tracking_cc.h.
The documentation for this class was generated from the following file:

- gps_l1_ca_gaussian_tracking_cc.h

12.192 Gps_L1_Ca_Tcp_Connector_Tracking_cc Class Reference

This class implements a DLL + PLL tracking loop block.
`#include <gps_l1_ca_tcp_connector_tracking_cc.h>`
Inheritance diagram for Gps_L1_Ca_Tcp_Connector_Tracking_cc:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro ∗p_gnss_synchro)
- void **start_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)

Friends

- gps_l1_ca_tcp_connector_tracking_cc_sptr **gps_l1_ca_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float early_late_space_chips, size_t port_ch0)

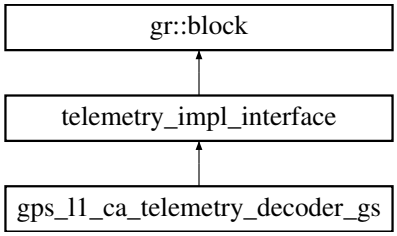
12.192.1 Detailed Description

This class implements a DLL + PLL tracking loop block.
Definition at line 58 of file gps_l1_ca_tcp_connector_tracking_cc.h.
The documentation for this class was generated from the following file:

- gps_l1_ca_tcp_connector_tracking_cc.h

12.193 gps_l1_ca_telemetry_decoder_gs Class Reference

This class implements a block that decodes the NAV data defined in IS-GPS-200M.
`#include <gps_l1_ca_telemetry_decoder_gs.h>`
Inheritance diagram for gps_l1_ca_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int channel) override

  Set receiver's channel.
- void reset () override
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)

Friends

- gps_l1_ca_telemetry_decoder_gs_sptr **gps_l1_ca_make_telemetry_decoder_gs** (const Tlm_Conf &conf, L1LnavSystem system=L1LnavSystem::GPS)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.193.1 Detailed Description

This class implements a block that decodes the NAV data defined in IS-GPS-200M.
Definition at line 53 of file gps_l1_ca_telemetry_decoder_gs.h.

12.193.2 Member Function Documentation

12.193.2.1 general_work()

```
int gps_l1_ca_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.193.2.2 reset()

```
void gps_l1_ca_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.193.2.3 set_channel()

```
void gps_l1_ca_telemetry_decoder_gs::set_channel (
            int channel)  [override], [virtual]
```

Set receiver's channel.
Implements telemetry_impl_interface.

12.193.2.4 set_satellite()

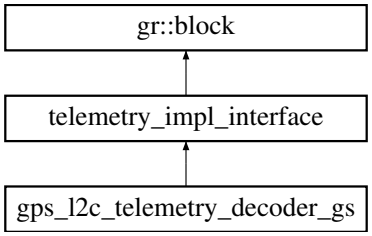
```
void gps_l1_ca_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- gps_l1_ca_telemetry_decoder_gs.h

12.194 gps_l2c_telemetry_decoder_gs Class Reference

This class implements a block that decodes CNAV data defined in IS-GPS-200M.
`#include <gps_l2c_telemetry_decoder_gs.h>`
Inheritance diagram for gps_l2c_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int32_t channel) override

  Set receiver's channel.
- void reset () override
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_channel (int channel)=0

Friends

- gps_l2c_telemetry_decoder_gs_sptr **gps_l2c_make_telemetry_decoder_gs** (const Tlm_Conf &conf)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.194.1 Detailed Description

This class implements a block that decodes CNAV data defined in IS-GPS-200M.
Definition at line 46 of file gps_l2c_telemetry_decoder_gs.h.

12.194.2 Member Function Documentation

12.194.2.1 general_work()

```
int gps_l2c_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.194.2.2 reset()

```
void gps_l2c_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.194.2.3 set_channel()

```
void gps_l2c_telemetry_decoder_gs::set_channel (
            int32_t channel)  [override]
```

Set receiver's channel.

12.194.2.4 set_satellite()

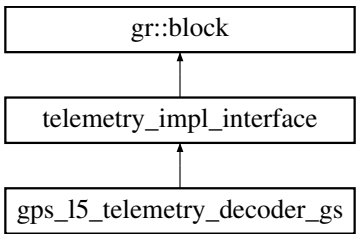
```
void gps_l2c_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- gps_l2c_telemetry_decoder_gs.h

12.195 gps_l5_telemetry_decoder_gs Class Reference

This class implements a GPS L5 Telemetry decoder.
`#include <gps_l5_telemetry_decoder_gs.h>`
Inheritance diagram for gps_l5_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int32_t channel) override

  Set receiver's channel.
- void reset () override

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_channel (int channel)=0

Friends

- gps_l5_telemetry_decoder_gs_sptr **gps_l5_make_telemetry_decoder_gs** (const Tlm_Conf &conf, Cnav↩ System system=CnavSystem::GPS)

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.195.1 Detailed Description

This class implements a GPS L5 Telemetry decoder.
Definition at line 50 of file gps_l5_telemetry_decoder_gs.h.

12.195.2 Member Function Documentation

12.195.2.1 reset()

```
void gps_l5_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.195.2.2 set_channel()

```
void gps_l5_telemetry_decoder_gs::set_channel (
            int32_t channel)  [override]
```

Set receiver's channel.

12.195.2.3 set_satellite()

```
void gps_l5_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- gps_l5_telemetry_decoder_gs.h

12.196 Gps_Navigation_Message Class Reference

This class decodes a GPS NAV Data message as described in IS-GPS-200M.
`#include <gps_navigation_message.h>`

Public Member Functions

- Gps_Navigation_Message (LnavSystem system=LnavSystem::GPS)
- Gps_Ephemeris get_ephemeris () const

  Obtain a GPS SV Ephemeris class filled with current SV data.
- Gps_Iono get_iono ()

  Obtain a GPS ionospheric correction parameters class filled with current SV data.
- Gps_Almanac get_almanac ()

  Obtain a GPS almanac class filled with current SV data.
- Gps_Utc_Model get_utc_model ()

  Obtain a GPS Almanac model parameters class filled with current SV data.
- int32_t subframe_decoder (const char *subframe)

  Decodes the GPS NAV message.
- double utc_time (double gpstime_corrected) const

  Computes the Coordinated Universal Time (UTC) and returns it in [s] (IS-GPS-200M, 20.3.3.5.2.4).
- int32_t get_TOW () const

  Gets Time of Week, in seconds.
- int32_t get_GPS_week () const

  Gets GPS Week Number.
- void set_satellite_PRN (uint32_t prn)

  Sets satellite PRN number.
- uint32_t get_satellite_PRN () const

  Gets satellite PRN number.
- void set_channel (int32_t channel_id)

  Sets channel ID.
- bool get_flag_iono_valid () const

  Gets flag_iono_valid.
- bool get_flag_utc_model_valid () const

  Gets flag_utc_model_valid.
- int32_t get_almanac_health (uint32_t prn) const

  Gets the almanac health field for a satellite PRN.
- int32_t get_almanac_config_code (uint32_t prn) const

  Gets the 4-bit A-S flag and SV configuration term (page 25 of subframe 4) of a satellite, or -1 if not decoded yet.
- bool **satellite_validation** ()
- bool **almanac_validation** () const

12.196.1 Detailed Description

This class decodes a GPS NAV Data message as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
Definition at line 52 of file gps_navigation_message.h.

12.196.2 Constructor & Destructor Documentation

12.196.2.1 Gps_Navigation_Message()

```
Gps_Navigation_Message::Gps_Navigation_Message (
            LnavSystem system = LnavSystem::GPS)  [explicit]
```

Default constructor

12.196.3 Member Function Documentation

12.196.3.1 get_almanac()

```
Gps_Almanac Gps_Navigation_Message::get_almanac ()
```

Obtain a GPS almanac class filled with current SV data.

12.196.3.2 get_almanac_config_code()

```
int32_t Gps_Navigation_Message::get_almanac_config_code (
            uint32_t prn) const
```

Gets the 4-bit A-S flag and SV configuration term (page 25 of subframe 4) of a satellite, or -1 if not decoded yet.

12.196.3.3 get_almanac_health()

```
int32_t Gps_Navigation_Message::get_almanac_health (
            uint32_t prn) const
```

Gets the almanac health field for a satellite PRN.

12.196.3.4 get_ephemeris()

```
Gps_Ephemeris Gps_Navigation_Message::get_ephemeris () const
```

Obtain a GPS SV Ephemeris class filled with current SV data.

12.196.3.5 get_flag_iono_valid()

```
bool Gps_Navigation_Message::get_flag_iono_valid () const  [inline]
```

Gets flag_iono_valid.
Definition at line 134 of file gps_navigation_message.h.

12.196.3.6 get_flag_utc_model_valid()

```
bool Gps_Navigation_Message::get_flag_utc_model_valid () const  [inline]
```

Gets flag_utc_model_valid.
Definition at line 142 of file gps_navigation_message.h.

12.196.3.7 get_GPS_week()

```
int32_t Gps_Navigation_Message::get_GPS_week () const  [inline]
```

Gets GPS Week Number.
Definition at line 102 of file gps_navigation_message.h.

12.196.3.8 get_iono()

```
Gps_Iono Gps_Navigation_Message::get_iono ()
```

Obtain a GPS ionospheric correction parameters class filled with current SV data.

12.196.3.9 get_satellite_PRN()

```
uint32_t Gps_Navigation_Message::get_satellite_PRN () const  [inline]
```

Gets satellite PRN number.
Definition at line 118 of file gps_navigation_message.h.

12.196.3.10 get_TOW()

```
int32_t Gps_Navigation_Message::get_TOW () const  [inline]
```

Gets Time of Week, in seconds.
Definition at line 94 of file gps_navigation_message.h.

12.196.3.11 get_utc_model()

```
Gps_Utc_Model Gps_Navigation_Message::get_utc_model ()
```

Obtain a GPS Almanac model parameters class filled with current SV data.

12.196.3.12 set_channel()

```
void Gps_Navigation_Message::set_channel (
            int32_t channel_id)  [inline]
```

Sets channel ID.
Definition at line 126 of file gps_navigation_message.h.

12.196.3.13 set_satellite_PRN()

```
void Gps_Navigation_Message::set_satellite_PRN (
            uint32_t prn)  [inline]
```

Sets satellite PRN number.
Definition at line 110 of file gps_navigation_message.h.

12.196.3.14 subframe_decoder()

```
int32_t Gps_Navigation_Message::subframe_decoder (
            const char * subframe)
```

Decodes the GPS NAV message.

12.196.3.15 utc_time()

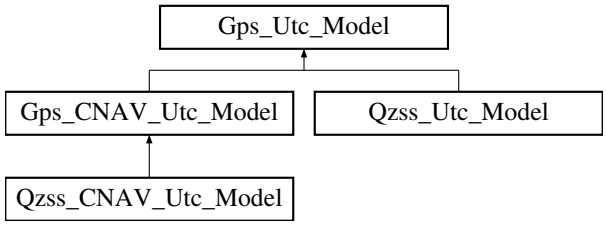
```
double Gps_Navigation_Message::utc_time (
            double gpstime_corrected) const
```

Computes the Coordinated Universal Time (UTC) and returns it in [s] (IS-GPS-200M, 20.3.3.5.2.4).
The documentation for this class was generated from the following file:

- gps_navigation_message.h

12.197 Gps_Utc_Model Class Reference

This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M.
`#include <gps_utc_model.h>`
Inheritance diagram for Gps_Utc_Model:



Public Member Functions

- Gps_Utc_Model ()=default
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

Public Attributes

- double A0 {}

  Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s].
- double A1 {}

  1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- double A2 {}

  2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]

- int32_t tot {}

  Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200M) [s].
- int32_t WN_T {}

  UTC reference week number [weeks].
- int32_t DeltaT_LS {}

  Delta time due to leap seconds [s]. Number of leap seconds since 6-Jan-1980 as transmitted by the GPS almanac.
- int32_t WN_LSF {}

  Week number at the end of which the leap second becomes effective [weeks].
- int32_t DN {}

  Day number (DN) at the end of which the leap second becomes effective [days].
- int32_t DeltaT_LSF {}

  Scheduled future or recent past (relative to NAV message upload) value of the delta time due to leap seconds [s].
- bool valid {}

12.197.1 Detailed Description

This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M.
See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
Definition at line 35 of file gps_utc_model.h.

12.197.2 Constructor & Destructor Documentation

12.197.2.1 Gps_Utc_Model()

```
Gps_Utc_Model::Gps_Utc_Model ()  [default]
```

Default constructor
Referenced by Qzss_Utc_Model::Qzss_Utc_Model().

12.197.3 Member Function Documentation

12.197.3.1 serialize()

```
template<class Archive>
void Gps_Utc_Model::serialize (
            Archive & archive,
            const uint32_t version)  [inline]
```

Definition at line 61 of file gps_utc_model.h.

12.197.4 Member Data Documentation

12.197.4.1 A0

```
double Gps_Utc_Model::A0 {}
```

Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s].
Definition at line 44 of file gps_utc_model.h.

12.197.4.2 A1

```
double Gps_Utc_Model::A1 {}
```

1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
Definition at line 45 of file gps_utc_model.h.

12.197.4.3 A2

```
double Gps_Utc_Model::A2 {}
```

2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
Definition at line 46 of file gps_utc_model.h.

12.197.4.4 DeltaT_LS

`int32_t Gps_Utc_Model::DeltaT_LS {}`
Delta time due to leap seconds [s]. Number of leap seconds since 6-Jan-1980 as transmitted by the GPS almanac.
Definition at line 49 of file gps_utc_model.h.

12.197.4.5 DeltaT_LSF

`int32_t Gps_Utc_Model::DeltaT_LSF {}`
Scheduled future or recent past (relative to NAV message upload) value of the delta time due to leap seconds [s].
Definition at line 52 of file gps_utc_model.h.

12.197.4.6 DN

`int32_t Gps_Utc_Model::DN {}`
Day number (DN) at the end of which the leap second becomes effective [days].
Definition at line 51 of file gps_utc_model.h.

12.197.4.7 tot

`int32_t Gps_Utc_Model::tot {}`
Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200M) [s].
Definition at line 47 of file gps_utc_model.h.

12.197.4.8 valid

`bool Gps_Utc_Model::valid {}`
Definition at line 54 of file gps_utc_model.h.

12.197.4.9 WN_LSF

`int32_t Gps_Utc_Model::WN_LSF {}`
Week number at the end of which the leap second becomes effective [weeks].
Definition at line 50 of file gps_utc_model.h.

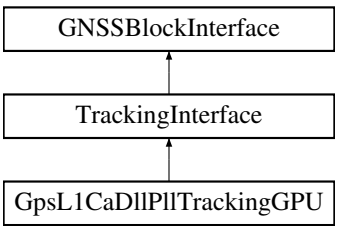
12.197.4.10 WN_T

`int32_t Gps_Utc_Model::WN_T {}`
UTC reference week number [weeks].
Definition at line 48 of file gps_utc_model.h.
The documentation for this class was generated from the following file:

- gps_utc_model.h

12.198 GpsL1CaDllPllTrackingGPU Class Reference

This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions.
`#include <gps_l1_ca_dll_pll_tracking_gpu.h>`
Inheritance diagram for GpsL1CaDllPllTrackingGPU:



Public Member Functions

- **GpsL1CaDllPllTrackingGPU** (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "GPS_L1_CA_DLL_PLL_Tracking_GPU".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override
- void stop_tracking () override

  Stop running tracking.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.198.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions.
Definition at line 41 of file gps_l1_ca_dll_pll_tracking_gpu.h.

12.198.2 Member Function Documentation

12.198.2.1 connect()

```
void GpsL1CaDllPllTrackingGPU::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.198.2.2 disconnect()

```
void GpsL1CaDllPllTrackingGPU::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.198.2.3 get_left_block()

```
gr::basic_block_sptr GpsL1CaDllPllTrackingGPU::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.198.2.4 get_right_block()

```
gr::basic_block_sptr GpsL1CaDllPllTrackingGPU::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.198.2.5 implementation()

```
std::string GpsL1CaDllPllTrackingGPU::implementation ()  [inline], [override], [virtual]
```
Returns "GPS_L1_CA_DLL_PLL_Tracking_GPU".
Implements GNSSBlockInterface.
Definition at line 58 of file gps_l1_ca_dll_pll_tracking_gpu.h.

12.198.2.6 item_size()

```
size_t GpsL1CaDllPllTrackingGPU::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 63 of file gps_l1_ca_dll_pll_tracking_gpu.h.

12.198.2.7 role()

```
std::string GpsL1CaDllPllTrackingGPU::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 52 of file gps_l1_ca_dll_pll_tracking_gpu.h.

12.198.2.8 set_channel()

```
void GpsL1CaDllPllTrackingGPU::set_channel (
            unsigned int channel)  [override], [virtual]
```
Set tracking channel unique ID.
Implements TrackingInterface.

12.198.2.9 set_gnss_synchro()

```
void GpsL1CaDllPllTrackingGPU::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```
Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.198.2.10 start_tracking()

```
void GpsL1CaDllPllTrackingGPU::start_tracking ()  [override], [virtual]
```
Implements TrackingInterface.

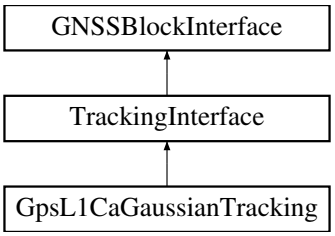
12.198.2.11 stop_tracking()

```
void GpsL1CaDllPllTrackingGPU::stop_tracking ()  [override], [virtual]
```
Stop running tracking.
Implements TrackingInterface.
The documentation for this class was generated from the following file:

- gps_l1_ca_dll_pll_tracking_gpu.h

12.199 GpsL1CaGaussianTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.
`#include <gps_l1_ca_gaussian_tracking.h>`
Inheritance diagram for GpsL1CaGaussianTracking:



Public Member Functions

- **GpsL1CaGaussianTracking** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "GPS_L1_CA_Gaussian_Tracking".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override
- void stop_tracking () override

  Stop running tracking.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.199.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.
Definition at line 44 of file gps_l1_ca_gaussian_tracking.h.

12.199.2 Member Function Documentation

12.199.2.1 connect()

```
void GpsL1CaGaussianTracking::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.199.2.2 disconnect()

```
void GpsL1CaGaussianTracking::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.199.2.3 get_left_block()

```
gr::basic_block_sptr GpsL1CaGaussianTracking::get_left_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.199.2.4 get_right_block()

```
gr::basic_block_sptr GpsL1CaGaussianTracking::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.199.2.5 implementation()

```
std::string GpsL1CaGaussianTracking::implementation ()  [inline], [override], [virtual]
```
Returns "GPS_L1_CA_Gaussian_Tracking".
Implements GNSSBlockInterface.
Definition at line 61 of file gps_l1_ca_gaussian_tracking.h.

12.199.2.6 item_size()

```
size_t GpsL1CaGaussianTracking::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 66 of file gps_l1_ca_gaussian_tracking.h.

12.199.2.7 role()

```
std::string GpsL1CaGaussianTracking::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 55 of file gps_l1_ca_gaussian_tracking.h.

12.199.2.8 set_channel()

```
void GpsL1CaGaussianTracking::set_channel (
            unsigned int channel)  [override], [virtual]
```
Set tracking channel unique ID.
Implements TrackingInterface.

12.199.2.9 set_gnss_synchro()

```
void GpsL1CaGaussianTracking::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```
Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.199.2.10 start_tracking()

```
void GpsL1CaGaussianTracking::start_tracking ()  [override], [virtual]
```
Implements TrackingInterface.

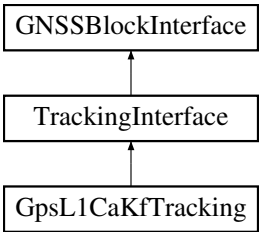
12.199.2.11 stop_tracking()

```
void GpsL1CaGaussianTracking::stop_tracking ()  [override], [virtual]
```
Stop running tracking.
Implements TrackingInterface.
The documentation for this class was generated from the following file:

- gps_l1_ca_gaussian_tracking.h

12.200 GpsL1CaKfTracking Class Reference

This class implements a code + carrier Kalman Filter tracking loop with VTL capabilities.
`#include <gps_l1_ca_kf_tracking.h>`
Inheritance diagram for GpsL1CaKfTracking:



Public Member Functions

- **GpsL1CaKfTracking** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "GPS_L1_CA_KF_Tracking".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override
- void stop_tracking () override

  Stop running tracking.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.200.1 Detailed Description

This class implements a code + carrier Kalman Filter tracking loop with VTL capabilities.
Definition at line 36 of file gps_l1_ca_kf_tracking.h.

12.200.2 Member Function Documentation

12.200.2.1 connect()

```
void GpsL1CaKfTracking::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.200.2.2 disconnect()

```
void GpsL1CaKfTracking::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.200.2.3 get_left_block()

```
gr::basic_block_sptr GpsL1CaKfTracking::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.200.2.4 get_right_block()

```
gr::basic_block_sptr GpsL1CaKfTracking::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.200.2.5 implementation()

```
std::string GpsL1CaKfTracking::implementation ()  [inline], [override], [virtual]
```

Returns "GPS_L1_CA_KF_Tracking".
Implements GNSSBlockInterface.
Definition at line 53 of file gps_l1_ca_kf_tracking.h.

12.200.2.6 item_size()

```
size_t GpsL1CaKfTracking::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 58 of file gps_l1_ca_kf_tracking.h.

12.200.2.7 role()

```
std::string GpsL1CaKfTracking::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 47 of file gps_l1_ca_kf_tracking.h.

12.200.2.8 set_channel()

```
void GpsL1CaKfTracking::set_channel (
            unsigned int channel)  [override], [virtual]
```

Set tracking channel unique ID.
Implements TrackingInterface.

12.200.2.9 set_gnss_synchro()

```
void GpsL1CaKfTracking::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.200.2.10 start_tracking()

```
void GpsL1CaKfTracking::start_tracking ()  [override], [virtual]
```

Implements TrackingInterface.

12.200.2.11 stop_tracking()

```
void GpsL1CaKfTracking::stop_tracking ()  [override], [virtual]
```

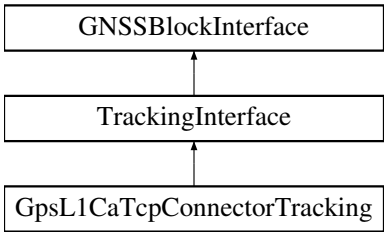
Stop running tracking.
Implements TrackingInterface.

The documentation for this class was generated from the following file:

- gps_l1_ca_kf_tracking.h

12.201 GpsL1CaTcpConnectorTracking Class Reference

This class implements a code DLL + carrier PLL tracking loop.
`#include <gps_l1_ca_tcp_connector_tracking.h>`
Inheritance diagram for GpsL1CaTcpConnectorTracking:



Public Member Functions

- **GpsL1CaTcpConnectorTracking** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "GPS_L1_CA_TCP_CONNECTOR_Tracking".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_channel (unsigned int channel) override

  Set tracking channel unique ID.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
- void start_tracking () override
- void stop_tracking () override

  Stop running tracking.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.201.1 Detailed Description

This class implements a code DLL + carrier PLL tracking loop.
Definition at line 42 of file gps_l1_ca_tcp_connector_tracking.h.

12.201.2 Member Function Documentation

12.201.2.1 connect()

```
void GpsL1CaTcpConnectorTracking::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.201.2.2 disconnect()

```
void GpsL1CaTcpConnectorTracking::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.201.2.3 get_left_block()

```
gr::basic_block_sptr GpsL1CaTcpConnectorTracking::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.201.2.4 get_right_block()

```
gr::basic_block_sptr GpsL1CaTcpConnectorTracking::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.201.2.5 implementation()

```
std::string GpsL1CaTcpConnectorTracking::implementation ()  [inline], [override], [virtual]
```

Returns "GPS_L1_CA_TCP_CONNECTOR_Tracking".
Implements GNSSBlockInterface.
Definition at line 59 of file gps_l1_ca_tcp_connector_tracking.h.

12.201.2.6 item_size()

```
size_t GpsL1CaTcpConnectorTracking::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 64 of file gps_l1_ca_tcp_connector_tracking.h.

12.201.2.7 role()

```
std::string GpsL1CaTcpConnectorTracking::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 53 of file gps_l1_ca_tcp_connector_tracking.h.

12.201.2.8 set_channel()

```
void GpsL1CaTcpConnectorTracking::set_channel (
            unsigned int channel)  [override], [virtual]
```

Set tracking channel unique ID.
Implements TrackingInterface.

12.201.2.9 set_gnss_synchro()

```
void GpsL1CaTcpConnectorTracking::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements TrackingInterface.

12.201.2.10 start_tracking()

```
void GpsL1CaTcpConnectorTracking::start_tracking ()  [override], [virtual]
```
Implements TrackingInterface.

12.201.2.11 stop_tracking()

```
void GpsL1CaTcpConnectorTracking::stop_tracking ()  [override], [virtual]
```
Stop running tracking.
Implements TrackingInterface.
The documentation for this class was generated from the following file:

- gps_l1_ca_tcp_connector_tracking.h

12.202 GPU_Complex Struct Reference

Public Member Functions

- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex (float a, float b)
- CUDA_CALLABLE_MEMBER_DEVICE float magnitude2 (void) const
- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex operator∗ (const GPU_Complex &a) const
- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex operator+ (const GPU_Complex &a) const
- CUDA_CALLABLE_MEMBER_DEVICE void operator+= (const GPU_Complex &a)
- CUDA_CALLABLE_MEMBER_DEVICE void multiply_acc (const GPU_Complex &a, const GPU_Complex &b)

Public Attributes

- float r
- float i

12.202.1 Detailed Description

Definition at line 45 of file cuda_multicorrelator.h.

12.202.2 Constructor & Destructor Documentation

12.202.2.1 GPU_Complex() [1/2]

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex::GPU_Complex ()  [inline]
```
Definition at line 49 of file cuda_multicorrelator.h.

12.202.2.2 GPU_Complex() [2/2]

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex::GPU_Complex (
            float a,
            float b)  [inline]
```
Definition at line 50 of file cuda_multicorrelator.h.

12.202.3 Member Function Documentation

12.202.3.1 magnitude2()

```
CUDA_CALLABLE_MEMBER_DEVICE float GPU_Complex::magnitude2 (
            void ) const  [inline]
```
Definition at line 51 of file cuda_multicorrelator.h.

12.202.3.2 multiply_acc()

```
CUDA_CALLABLE_MEMBER_DEVICE void GPU_Complex::multiply_acc (
            const GPU_Complex & a,
            const GPU_Complex & b)  [inline]
```

Definition at line 69 of file cuda_multicorrelator.h.

12.202.3.3 operator∗()

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex GPU_Complex::operator* (
            const GPU_Complex & a) const  [inline]
```

Definition at line 52 of file cuda_multicorrelator.h.

12.202.3.4 operator+()

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex GPU_Complex::operator+ (
            const GPU_Complex & a) const  [inline]
```

Definition at line 60 of file cuda_multicorrelator.h.

12.202.3.5 operator+=()

```
CUDA_CALLABLE_MEMBER_DEVICE void GPU_Complex::operator+= (
            const GPU_Complex & a)  [inline]
```

Definition at line 64 of file cuda_multicorrelator.h.

12.202.4 Member Data Documentation

12.202.4.1 i

```
float GPU_Complex::i
```

Definition at line 48 of file cuda_multicorrelator.h.

12.202.4.2 r

```
float GPU_Complex::r
```

Definition at line 47 of file cuda_multicorrelator.h.
The documentation for this struct was generated from the following file:

- cuda_multicorrelator.h

12.203 GPU_Complex_Short Struct Reference

Public Member Functions

- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short (short int a, short int b)
- CUDA_CALLABLE_MEMBER_DEVICE float magnitude2 (void) const
- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short operator∗ (const GPU_Complex_Short &a) const
- CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short operator+ (const GPU_Complex_Short &a) const

Public Attributes

- float r
- float i

12.203.1 Detailed Description

Definition at line 88 of file cuda_multicorrelator.h.

12.203.2 Constructor & Destructor Documentation

12.203.2.1 GPU_Complex_Short()

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short::GPU_Complex_Short (
            short int a,
            short int b)  [inline]
```

Definition at line 92 of file cuda_multicorrelator.h.

12.203.3 Member Function Documentation

12.203.3.1 magnitude2()

```
CUDA_CALLABLE_MEMBER_DEVICE float GPU_Complex_Short::magnitude2 (
            void ) const  [inline]
```

Definition at line 93 of file cuda_multicorrelator.h.

12.203.3.2 operator∗()

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short GPU_Complex_Short::operator* (
            const GPU_Complex_Short & a) const  [inline]
```

Definition at line 97 of file cuda_multicorrelator.h.

12.203.3.3 operator+()

```
CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short GPU_Complex_Short::operator+ (
            const GPU_Complex_Short & a) const  [inline]
```

Definition at line 101 of file cuda_multicorrelator.h.

12.203.4 Member Data Documentation

12.203.4.1 i

```
float GPU_Complex_Short::i
```

Definition at line 91 of file cuda_multicorrelator.h.

12.203.4.2 r

```
float GPU_Complex_Short::r
```

Definition at line 90 of file cuda_multicorrelator.h.

The documentation for this struct was generated from the following file:

- cuda_multicorrelator.h

12.204 Gpx_Printer Class Reference

Prints PVT information to GPX format file.

```
#include <gpx_printer.h>
```

Public Member Functions

- **Gpx_Printer** (const std::string &base_path=".")
- bool **set_headers** (const std::string &filename, bool time_tag_name=true)
- bool **print_position** (const Pvt_Solution ∗const position)
- bool **close_file** ()

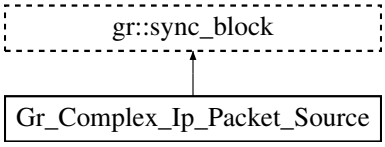
12.204.1 Detailed Description

Prints PVT information to GPX format file.
See https://www.topografix.com/gpx.asp
Definition at line 39 of file gpx_printer.h.
The documentation for this class was generated from the following file:

- gpx_printer.h

12.205 Gr_Complex_Ip_Packet_Source Class Reference

Inheritance diagram for Gr_Complex_Ip_Packet_Source:



Public Types

- using sptr = gnss_shared_ptr<Gr_Complex_Ip_Packet_Source>

Public Member Functions

- **Gr_Complex_Ip_Packet_Source** (std::string src_device, const std::string &origin_address, int udp_port, int udp_packet_size, int n_baseband_channels, const std::string &wire_sample_type, size_t item_size, bool IQ↩_swap_)
- bool **start** ()
- bool **stop** ()
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Static Public Member Functions

- static sptr **make** (std::string src_device, const std::string &origin_address, int udp_port, int udp_packet_size, int n_baseband_channels, const std::string &wire_sample_type, size_t item_size, bool IQ_swap_)

12.205.1 Detailed Description

Definition at line 41 of file gr_complex_ip_packet_source.h.

12.205.2 Member Typedef Documentation

12.205.2.1 sptr

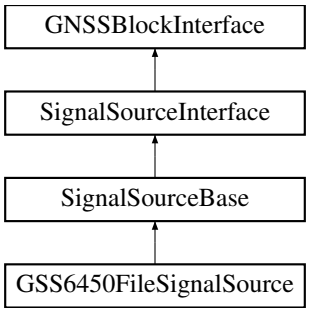
```
using Gr_Complex_Ip_Packet_Source::sptr = gnss_shared_ptr<Gr_Complex_Ip_Packet_Source>
```

Definition at line 44 of file gr_complex_ip_packet_source.h.
The documentation for this class was generated from the following file:

- gr_complex_ip_packet_source.h

12.206 GSS6450FileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <gss6450_file_signal_source.h>`
Inheritance diagram for GSS6450FileSignalSource:



Public Member Functions

- **GSS6450FileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, uint32_t in_streams, uint32_t out_streams, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override
- gr::basic_block_sptr get_right_block () override
- size_t getRfChannels () const override
- std::string filename () const
- std::string item_type () const
- bool repeat () const
- int64_t sampling_frequency () const
- uint64_t samples () const

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.206.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 52 of file gss6450_file_signal_source.h.

12.206.2 Member Function Documentation

12.206.2.1 connect()

```
void GSS6450FileSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.206.2.2 disconnect()

```
void GSS6450FileSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.206.2.3 filename()

```
std::string GSS6450FileSignalSource::filename () const  [inline]
```

Definition at line 70 of file gss6450_file_signal_source.h.

12.206.2.4 get_left_block()

```
gr::basic_block_sptr GSS6450FileSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.206.2.5 get_right_block() [1/2]

```
gr::basic_block_sptr GSS6450FileSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.206.2.6 get_right_block() [2/2]

```
gr::basic_block_sptr GSS6450FileSignalSource::get_right_block (
            int RF_channel)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.206.2.7 getRfChannels()

```
size_t GSS6450FileSignalSource::getRfChannels () const  [override], [virtual]
```

Implements SignalSourceInterface.

12.206.2.8 item_size()

```
size_t GSS6450FileSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 58 of file gss6450_file_signal_source.h.

12.206.2.9 item_type()

```
std::string GSS6450FileSignalSource::item_type () const  [inline]
```

Definition at line 75 of file gss6450_file_signal_source.h.

12.206.2.10 repeat()

```
bool GSS6450FileSignalSource::repeat () const  [inline]
```

Definition at line 80 of file gss6450_file_signal_source.h.

12.206.2.11 samples()

```
uint64_t GSS6450FileSignalSource::samples () const  [inline]
```

Definition at line 90 of file gss6450_file_signal_source.h.

12.206.2.12 sampling_frequency()

```
int64_t GSS6450FileSignalSource::sampling_frequency () const  [inline]
```
Definition at line 85 of file gss6450_file_signal_source.h.
The documentation for this class was generated from the following file:

- gss6450_file_signal_source.h

12.207 gtime_t Struct Reference

Public Attributes

- time_t time
- double sec

12.207.1 Detailed Description

Definition at line 382 of file rtklib.h.

12.207.2 Member Data Documentation

12.207.2.1 sec

```
double gtime_t::sec
```
Definition at line 385 of file rtklib.h.

12.207.2.2 time

```
time_t gtime_t::time
```
Definition at line 384 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.208 half_cyc_tag Struct Reference

Public Attributes

- unsigned char sat
- unsigned char freq
- unsigned char valid
- char corr
- gtime_t ts
- gtime_t te
- struct half_cyc_tag * next

12.208.1 Detailed Description

Definition at line 1149 of file rtklib.h.

12.208.2 Member Data Documentation

12.208.2.1 corr

```
char half_cyc_tag::corr
```
Definition at line 1154 of file rtklib.h.

12.208.2.2 freq

```
unsigned char half_cyc_tag::freq
```
Definition at line 1152 of file rtklib.h.

12.208.2.3 next

```
struct half_cyc_tag* half_cyc_tag::next
```
Definition at line 1156 of file rtklib.h.

12.208.2.4 sat

```
unsigned char half_cyc_tag::sat
```
Definition at line 1151 of file rtklib.h.

12.208.2.5 te

```
gtime_t half_cyc_tag::te
```
Definition at line 1155 of file rtklib.h.

12.208.2.6 ts

```
gtime_t half_cyc_tag::ts
```
Definition at line 1155 of file rtklib.h.

12.208.2.7 valid

```
unsigned char half_cyc_tag::valid
```
Definition at line 1153 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.209 HAS_bias_corrections Class Reference

Public Attributes

- float bias {}
- uint32_t valid_until {}
- uint8_t mask_id {}
- uint8_t iod_set_id {}

12.209.1 Detailed Description

Definition at line 68 of file rtklib_conversions.h.

12.209.2 Member Data Documentation

12.209.2.1 bias

```
float HAS_bias_corrections::bias {}
```
Definition at line 72 of file rtklib_conversions.h.

12.209.2.2 iod_set_id

```
uint8_t HAS_bias_corrections::iod_set_id {}
```
Definition at line 75 of file rtklib_conversions.h.

12.209.2.3 mask_id

```
uint8_t HAS_bias_corrections::mask_id {}
```
Definition at line 74 of file rtklib_conversions.h.

12.209.2.4 valid_until

```
uint32_t HAS_bias_corrections::valid_until {}
```
Definition at line 73 of file rtklib_conversions.h.
The documentation for this class was generated from the following file:

- rtklib_conversions.h

12.210 HAS_clock_corrections Class Reference

Public Attributes

- float clock_correction_m {}
- uint32_t valid_until {}
- uint16_t iod {}
- uint8_t mask_id {}
- uint8_t iod_set_id {}

12.210.1 Detailed Description

Definition at line 44 of file rtklib_conversions.h.

12.210.2 Member Data Documentation

12.210.2.1 clock_correction_m

```
float HAS_clock_corrections::clock_correction_m {}
```
Definition at line 48 of file rtklib_conversions.h.

12.210.2.2 iod

```
uint16_t HAS_clock_corrections::iod {}
```
Definition at line 50 of file rtklib_conversions.h.

12.210.2.3 iod_set_id

```
uint8_t HAS_clock_corrections::iod_set_id {}
```
Definition at line 52 of file rtklib_conversions.h.

12.210.2.4 mask_id

```
uint8_t HAS_clock_corrections::mask_id {}
```
Definition at line 51 of file rtklib_conversions.h.

12.210.2.5 valid_until

```
uint32_t HAS_clock_corrections::valid_until {}
```
Definition at line 49 of file rtklib_conversions.h.
The documentation for this class was generated from the following file:

- rtklib_conversions.h

12.211 HAS_obs_corrections Class Reference

Public Attributes

- std::string signal
- float code_bias_m {}
- float phase_bias_cycle {}
- uint8_t phase_discontinuity_indicator {}
- bool phase_bias_discontinuity {}

12.211.1 Detailed Description

Definition at line 78 of file rtklib_conversions.h.

12.211.2 Member Data Documentation

12.211.2.1 code_bias_m

```
float HAS_obs_corrections::code_bias_m {}
```

Definition at line 83 of file rtklib_conversions.h.

12.211.2.2 phase_bias_cycle

```
float HAS_obs_corrections::phase_bias_cycle {}
```

Definition at line 84 of file rtklib_conversions.h.

12.211.2.3 phase_bias_discontinuity

```
bool HAS_obs_corrections::phase_bias_discontinuity {}
```

Definition at line 86 of file rtklib_conversions.h.

12.211.2.4 phase_discontinuity_indicator

```
uint8_t HAS_obs_corrections::phase_discontinuity_indicator {}
```

Definition at line 85 of file rtklib_conversions.h.

12.211.2.5 signal

```
std::string HAS_obs_corrections::signal
```

Definition at line 82 of file rtklib_conversions.h.
The documentation for this class was generated from the following file:

- rtklib_conversions.h

12.212 HAS_orbit_corrections Class Reference

Public Attributes

- float radial_m {}
- float in_track_m {}
- float cross_track_m {}
- uint32_t valid_until {}
- uint16_t iod {}
- uint8_t mask_id {}
- uint8_t iod_set_id {}

12.212.1 Detailed Description

Definition at line 55 of file rtklib_conversions.h.

12.212.2 Member Data Documentation

12.212.2.1 cross_track_m

```
float HAS_orbit_corrections::cross_track_m {}
```
Definition at line 61 of file rtklib_conversions.h.

12.212.2.2 in_track_m

```
float HAS_orbit_corrections::in_track_m {}
```
Definition at line 60 of file rtklib_conversions.h.

12.212.2.3 iod

```
uint16_t HAS_orbit_corrections::iod {}
```
Definition at line 63 of file rtklib_conversions.h.

12.212.2.4 iod_set_id

```
uint8_t HAS_orbit_corrections::iod_set_id {}
```
Definition at line 65 of file rtklib_conversions.h.

12.212.2.5 mask_id

```
uint8_t HAS_orbit_corrections::mask_id {}
```
Definition at line 64 of file rtklib_conversions.h.

12.212.2.6 radial_m

```
float HAS_orbit_corrections::radial_m {}
```
Definition at line 59 of file rtklib_conversions.h.

12.212.2.7 valid_until

```
uint32_t HAS_orbit_corrections::valid_until {}
```
Definition at line 62 of file rtklib_conversions.h.
The documentation for this class was generated from the following file:

- rtklib_conversions.h

12.213 Has_Simple_Printer Class Reference

Prints HAS messages content in a txt file. See HAS-SIS-ICD for a message description.
`#include <has_simple_printer.h>`

Public Member Functions

- **Has_Simple_Printer** (const std::string &base_path=std::string("."), const std::string &filename=std::↩ string("HAS_Messages"), bool time_tag_name=true)
- bool **print_message** (const Galileo_HAS_data ∗const has_data)

12.213.1 Detailed Description

Prints HAS messages content in a txt file. See HAS-SIS-ICD for a message description.
Definition at line 37 of file has_simple_printer.h.
The documentation for this class was generated from the following file:

- has_simple_printer.h

12.214 std::hash< SensorIdentifier::value_type > Struct Reference

Public Member Functions

- std::size_t operator() (const SensorIdentifier::value_type &key) const noexcept

12.214.1 Detailed Description

Definition at line 67 of file sensor_identifier.h.

12.214.2 Member Function Documentation

12.214.2.1 operator()()

```
std::size_t std::hash< SensorIdentifier::value_type >::operator() (
            const SensorIdentifier::value_type & key) const  [inline], [noexcept]
```

Definition at line 69 of file sensor_identifier.h.
The documentation for this struct was generated from the following file:

- sensor_identifier.h

12.215 HistogramBitSynchronizer Class Reference

Histogram-based navigation data bit synchronizer.
`#include <bit_synchronizer.h>`

Classes

- struct Config

  Configuration parameters for HistogramBitSynchronizer.

Public Member Functions

- HistogramBitSynchronizer (const Config &cfg)

  Construct a histogram bit synchronizer with the provided configuration.
- void reset ()

  Reset the synchronizer state.
- bool update (const std::complex< float > &prompt, bool tracking_quality_ok)

  Update the synchronizer once per epoch.
- bool locked () const

  Query whether the synchronizer has achieved lock.
- int edge_phase () const

  Get the estimated bit edge phase bin.
- bool is_edge_epoch (std::int64_t k) const

  Predict whether a given epoch index corresponds to a bit edge.
- int bins () const

  Return the number of histogram bins.
- const std::vector< int > & get_histogram () const

  Access the internal histogram (read-only).
- std::int64_t get_total_events () const

  Total number of detected transition events accumulated into the histogram.
- std::int64_t get_epoch_count () const

  Total number of epochs processed by update().
- int epochs_until_next_edge () const

  Return the number of epochs until the next predicted navigation bit edge.

12.215.1 Detailed Description

Histogram-based navigation data bit synchronizer.
Definition at line 35 of file bit_synchronizer.h.

12.215.2 Constructor & Destructor Documentation

12.215.2.1 HistogramBitSynchronizer()

```
HistogramBitSynchronizer::HistogramBitSynchronizer (
            const Config & cfg)  [inline], [explicit]
```

Construct a histogram bit synchronizer with the provided configuration.
Initializes internal counters and allocates the histogram with bins() entries, all set to zero.

Parameters

cfg	Configuration parameters.

Definition at line 193 of file bit_synchronizer.h.
References bins().

12.215.3 Member Function Documentation

12.215.3.1 bins()

```
int HistogramBitSynchronizer::bins () const
```

Return the number of histogram bins.
Derived from the bit period and epoch duration, e.g.: bins = bit_period_ms / epoch_ms

Returns

Number of histogram bins.

Referenced by HistogramBitSynchronizer().

12.215.3.2 edge_phase()

```
int HistogramBitSynchronizer::edge_phase () const  [inline]
```

Get the estimated bit edge phase bin.
The edge phase is expressed as an integer histogram bin index in the range [0, bins()-1] when locked. The interpretation is "which epoch phase within the bit period is most likely to contain a navigation bit transition."

Returns

Estimated edge phase bin index, or -1 if not locked.

Definition at line 259 of file bit_synchronizer.h.

12.215.3.3 epochs_until_next_edge()

```
int HistogramBitSynchronizer::epochs_until_next_edge () const
```

Return the number of epochs until the next predicted navigation bit edge.
When the synchronizer is locked, this function computes the forward distance (in epochs) from the most recently processed epoch to the next epoch that is aligned with the estimated bit-edge phase.
The result is expressed modulo the bit period and has the following meaning:

- 0 : the current epoch corresponds to the predicted start of a new navigation bit
- $>$0 : number of epochs remaining until the next bit boundary

The computation is based on the internal epoch counter advanced by update(), assuming that update() is called once per epoch with a constant cadence equal to Config::epoch_ms.
If the synchronizer is not locked, or if the configuration yields an invalid number of bins, this function returns -1.

Returns

Number of epochs until the next predicted bit edge, or -1 if not locked or if the bit period configuration is invalid.

12.215.3.4 get_epoch_count()

```
std::int64_t HistogramBitSynchronizer::get_epoch_count () const  [inline]
```

Total number of epochs processed by update().
This counter increments once per call to update(), regardless of whether a transition is detected or whether tracking_quality_ok is true.

Returns

Total processed epochs.

Definition at line 310 of file bit_synchronizer.h.

12.215.3.5 get_histogram()

```
const std::vector< int > & HistogramBitSynchronizer::get_histogram () const  [inline]
```

Access the internal histogram (read-only).
Each entry counts how many detected candidate transitions occurred at the corresponding phase bin within the bit period.

Returns

Reference to the histogram vector.

Definition at line 293 of file bit_synchronizer.h.

12.215.3.6 get_total_events()

```
std::int64_t HistogramBitSynchronizer::get_total_events () const  [inline]
```

Total number of detected transition events accumulated into the histogram.

Returns

Total detected events.

Definition at line 300 of file bit_synchronizer.h.

12.215.3.7 is_edge_epoch()

```
bool HistogramBitSynchronizer::is_edge_epoch (
            std::int64_t k) const
```

Predict whether a given epoch index corresponds to a bit edge.
For a given epoch index `k` (0-based), this function returns true when `k` is aligned with the currently estimated edge phase (i.e., the predicted transition epoch), and false otherwise.
If not locked, this always returns false.

Parameters

k	Epoch index (0-based, consistent with the caller's epoch counting).

Returns

True if `k` is the predicted edge epoch; false otherwise.

12.215.3.8 locked()

```
bool HistogramBitSynchronizer::locked () const  [inline]
```

Query whether the synchronizer has achieved lock.

Returns

True if lock has been declared, false otherwise.

Definition at line 248 of file bit_synchronizer.h.

12.215.3.9 reset()

```
void HistogramBitSynchronizer::reset ()
```

Reset the synchronizer state.
Clears the histogram and all internal counters/flags, returning the instance to the pre-lock state:

- locked() becomes false
- edge_phase() becomes -1
- total_events and epoch_count are reset to zero

12.215.3.10 update()

```
bool HistogramBitSynchronizer::update (
            const std::complex< float > & prompt,
            bool tracking_quality_ok)
```

Update the synchronizer once per epoch.
This method should be called at a fixed cadence defined by Config::epoch_ms
The method:

- Advances the internal epoch counter,
- Optionally performs candidate transition detection if tracking quality is acceptable,
- Updates the phase histogram on detected transitions,
- Evaluates lock once enough events have been gathered.

Parameters

prompt	Prompt correlator output for the current epoch.
tracking_quality_ok	Indicates whether tracking quality is sufficient to trust the prompt sample for transition detection (e.g., code/carrier lock metrics).

Returns

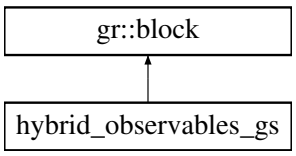
True only on the epoch when lock is first declared; false otherwise (including subsequent epochs after lock has been achieved).

The documentation for this class was generated from the following file:

- bit_synchronizer.h

12.216 hybrid_observables_gs Class Reference

This class implements a block that computes observables.
`#include <hybrid_observables_gs.h>`
Inheritance diagram for hybrid_observables_gs:



Public Member Functions

- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- hybrid_observables_gs_sptr **hybrid_observables_gs_make** (const Obs_Conf &conf_)

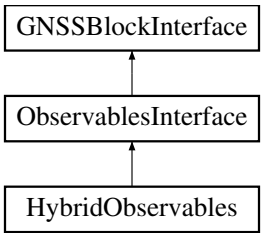
12.216.1 Detailed Description

This class implements a block that computes observables.
Definition at line 59 of file hybrid_observables_gs.h.
The documentation for this class was generated from the following file:

- hybrid_observables_gs.h

12.217 HybridObservables Class Reference

This class implements an ObservablesInterface for observables of all kind of GNSS signals.
`#include <hybrid_observables.h>`
Inheritance diagram for HybridObservables:



Public Member Functions

- **HybridObservables** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Hybrid_Observables".
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void reset () override
- size_t item_size () override

  All blocks must have an item_size() function implementation.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.217.1 Detailed Description

This class implements an ObservablesInterface for observables of all kind of GNSS signals.
Definition at line 43 of file hybrid_observables.h.

12.217.2 Member Function Documentation

12.217.2.1 connect()

```
void HybridObservables::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.217.2.2 disconnect()

```
void HybridObservables::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.217.2.3 get_left_block()

```
gr::basic_block_sptr HybridObservables::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.217.2.4 get_right_block()

```
gr::basic_block_sptr HybridObservables::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.217.2.5 implementation()

```
std::string HybridObservables::implementation ()  [inline], [override], [virtual]
```

Returns "Hybrid_Observables".
Implements GNSSBlockInterface.
Definition at line 59 of file hybrid_observables.h.

12.217.2.6 item_size()

```
size_t HybridObservables::item_size ()  [inline], [override], [virtual]
```

All blocks must have an item_size() function implementation.
Implements GNSSBlockInterface.
Definition at line 75 of file hybrid_observables.h.

12.217.2.7 reset()

```
void HybridObservables::reset ()  [inline], [override], [virtual]
```

Implements ObservablesInterface.
Definition at line 69 of file hybrid_observables.h.

12.217.2.8 role()

```
std::string HybridObservables::role ()  [inline], [override], [virtual]
```

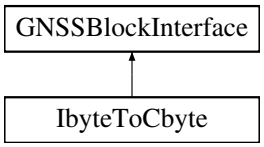
Implements GNSSBlockInterface.
Definition at line 53 of file hybrid_observables.h.
The documentation for this class was generated from the following file:

- hybrid_observables.h

12.218 IbyteToCbyte Class Reference

```
#include <ibyte_to_cbyte.h>
```

Inheritance diagram for IbyteToCbyte:



Public Member Functions

- **IbyteToCbyte** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Ibyte_To_Cbyte".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.218.1 Detailed Description

\briefAdapts an I/Q interleaved byte (unsigned char) sample stream into a std::complex<unsigned char> stream
Definition at line 40 of file ibyte_to_cbyte.h.

12.218.2 Member Function Documentation

12.218.2.1 connect()

```
void IbyteToCbyte::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.218.2.2 disconnect()

```
void IbyteToCbyte::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.218.2.3 get_left_block()

`gr::basic_block_sptr IbyteToCbyte::get_left_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.218.2.4 get_right_block()

`gr::basic_block_sptr IbyteToCbyte::get_right_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.218.2.5 implementation()

`std::string IbyteToCbyte::implementation () [inline], [override], [virtual]`
Returns "Ibyte_To_Cbyte".
Implements GNSSBlockInterface.
Definition at line 55 of file ibyte_to_cbyte.h.

12.218.2.6 item_size()

`size_t IbyteToCbyte::item_size () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 60 of file ibyte_to_cbyte.h.

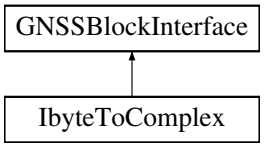
12.218.2.7 role()

`std::string IbyteToCbyte::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 49 of file ibyte_to_cbyte.h.
The documentation for this class was generated from the following file:

- ibyte_to_cbyte.h

12.219 IbyteToComplex Class Reference

Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.
`#include <ibyte_to_complex.h>`
Inheritance diagram for IbyteToComplex:



Public Member Functions

- **IbyteToComplex** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Ibyte_To_Complex".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.219.1 Detailed Description

Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.
Definition at line 39 of file ibyte_to_complex.h.

12.219.2 Member Function Documentation

12.219.2.1 connect()

```
void IbyteToComplex::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.219.2.2 disconnect()

```
void IbyteToComplex::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.219.2.3 get_left_block()

```
gr::basic_block_sptr IbyteToComplex::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.219.2.4 get_right_block()

```
gr::basic_block_sptr IbyteToComplex::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.219.2.5 implementation()

```
std::string IbyteToComplex::implementation ()  [inline], [override], [virtual]
```

Returns "Ibyte_To_Complex".
Implements GNSSBlockInterface.
Definition at line 54 of file ibyte_to_complex.h.

12.219.2.6 item_size()

```
size_t IbyteToComplex::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 59 of file ibyte_to_complex.h.

12.219.2.7 role()

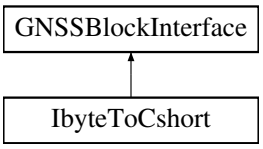
```
std::string IbyteToComplex::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 48 of file ibyte_to_complex.h.
The documentation for this class was generated from the following file:

- ibyte_to_complex.h

12.220 IbyteToCshort Class Reference

Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.
`#include <ibyte_to_cshort.h>`
Inheritance diagram for IbyteToCshort:



Public Member Functions

- **IbyteToCshort** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Ibyte_To_Cshort".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.220.1 Detailed Description

Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.
Definition at line 39 of file ibyte_to_cshort.h.

12.220.2 Member Function Documentation

12.220.2.1 connect()

```
void IbyteToCshort::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.220.2.2 disconnect()

```
void IbyteToCshort::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.220.2.3 get_left_block()

```
gr::basic_block_sptr IbyteToCshort::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.220.2.4 get_right_block()

```
gr::basic_block_sptr IbyteToCshort::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.220.2.5 implementation()

```
std::string IbyteToCshort::implementation ()  [inline], [override], [virtual]
```
Returns "Ibyte_To_Cshort".
Implements GNSSBlockInterface.
Definition at line 54 of file ibyte_to_cshort.h.

12.220.2.6 item_size()

```
size_t IbyteToCshort::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 59 of file ibyte_to_cshort.h.

12.220.2.7 role()

```
std::string IbyteToCshort::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 48 of file ibyte_to_cshort.h.
The documentation for this class was generated from the following file:

- ibyte_to_cshort.h

12.221 INIReader Class Reference

Read an INI file into easy-to-access name/value pairs. (Note that I've gone for simplicity here rather than speed, but it should be pretty decent.).
`#include <INIReader.h>`

Public Member Functions

- INIReader (const std::string &filename)

  Construct INIReader and parse given filename. See ini.h for more info about the parsing.
- int ParseError () const

  Return the result of ini_parse(), i.e., 0 on success, line number of first error on parse error, or -1 on file open error.
- std::string Get (const std::string &section, const std::string &name, const std::string &default_value)

  Get a string value from INI file, returning default_value if not found.
- int64_t GetInteger (const std::string &section, const std::string &name, int64_t default_value)

  Get an integer (long) value from INI file, returning default_value if not found.
- bool HasSection (const std::string &section) const

  Return true if the given section exists (section must contain at least one name=value pair).
- bool HasValue (const std::string &section, const std::string &name) const

  Return true if a value exists with the given section and field names.

12.221.1 Detailed Description

Read an INI file into easy-to-access name/value pairs. (Note that I've gone for simplicity here rather than speed, but it should be pretty decent.).
Definition at line 45 of file INIReader.h.

12.221.2 Constructor & Destructor Documentation

12.221.2.1 INIReader()

```
INIReader::INIReader (
            const std::string & filename)  [explicit]
```

Construct INIReader and parse given filename. See ini.h for more info about the parsing.

12.221.3 Member Function Documentation

12.221.3.1 Get()

```
std::string INIReader::Get (
            const std::string & section,
            const std::string & name,
            const std::string & default_value)
```

Get a string value from INI file, returning default_value if not found.

12.221.3.2 GetInteger()

```
int64_t INIReader::GetInteger (
            const std::string & section,
            const std::string & name,
            int64_t default_value)
```

Get an integer (long) value from INI file, returning default_value if not found.

12.221.3.3 HasSection()

```
bool INIReader::HasSection (
            const std::string & section) const
```

Return true if the given section exists (section must contain at least one name=value pair).

12.221.3.4 HasValue()

```
bool INIReader::HasValue (
            const std::string & section,
            const std::string & name) const
```

Return true if a value exists with the given section and field names.

12.221.3.5 ParseError()

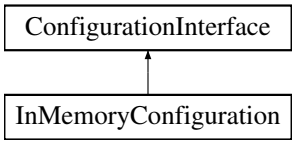
```
int INIReader::ParseError () const
```

Return the result of ini_parse(), i.e., 0 on success, line number of first error on parse error, or -1 on file open error.
The documentation for this class was generated from the following file:

- INIReader.h

12.222 InMemoryConfiguration Class Reference

This class is an implementation of the interface ConfigurationInterface.
`#include <in_memory_configuration.h>`
Inheritance diagram for InMemoryConfiguration:



Public Member Functions

- std::string property (std::string property_name, std::string default_value) const override
- bool property (std::string property_name, bool default_value) const override
- int64_t property (std::string property_name, int64_t default_value) const override
- uint64_t property (std::string property_name, uint64_t default_value) const override
- int32_t property (std::string property_name, int32_t default_value) const override
- uint32_t property (std::string property_name, uint32_t default_value) const override
- int16_t property (std::string property_name, int16_t default_value) const override
- uint16_t property (std::string property_name, uint16_t default_value) const override
- float property (std::string property_name, float default_value) const override
- double property (std::string property_name, double default_value) const override
- void set_property (std::string property_name, std::string value) override
- void **supersede_property** (const std::string &property_name, const std::string &value)
- bool is_present (const std::string &property_name) const override

12.222.1 Detailed Description

This class is an implementation of the interface ConfigurationInterface.
This implementation accepts configuration parameters upon instantiation and it is intended to be used in unit testing.
Definition at line 43 of file in_memory_configuration.h.

12.222.2 Member Function Documentation

12.222.2.1 is_present()

```
bool InMemoryConfiguration::is_present (
            const std::string & property_name) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.2 property() [1/10]

```
bool InMemoryConfiguration::property (
            std::string property_name,
            bool default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.3 property() [2/10]

```
double InMemoryConfiguration::property (
            std::string property_name,
            double default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.4 property() [3/10]

```
float InMemoryConfiguration::property (
            std::string property_name,
            float default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.5 property() [4/10]

```
int16_t InMemoryConfiguration::property (
            std::string property_name,
            int16_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.6 property() [5/10]

```
int32_t InMemoryConfiguration::property (
            std::string property_name,
            int32_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.7 property() [6/10]

```
int64_t InMemoryConfiguration::property (
            std::string property_name,
            int64_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.8 property() [7/10]

```
std::string InMemoryConfiguration::property (
            std::string property_name,
            std::string default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.9 property() [8/10]

```
uint16_t InMemoryConfiguration::property (
            std::string property_name,
            uint16_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.10 property() [9/10]

```
uint32_t InMemoryConfiguration::property (
            std::string property_name,
            uint32_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.11 property() [10/10]

```
uint64_t InMemoryConfiguration::property (
            std::string property_name,
            uint64_t default_value) const  [override], [virtual]
```

Implements ConfigurationInterface.

12.222.2.12 set_property()

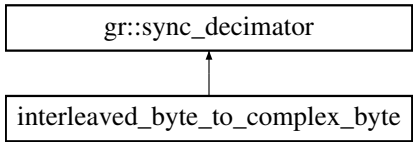
```
void InMemoryConfiguration::set_property (
            std::string property_name,
            std::string value)  [override], [virtual]
```

Implements ConfigurationInterface.
The documentation for this class was generated from the following file:

- in_memory_configuration.h

12.223 interleaved_byte_to_complex_byte Class Reference

This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::complex<unsigned char>).
`#include <interleaved_byte_to_complex_byte.h>`
Inheritance diagram for interleaved_byte_to_complex_byte:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- interleaved_byte_to_complex_byte_sptr **make_interleaved_byte_to_complex_byte** ()

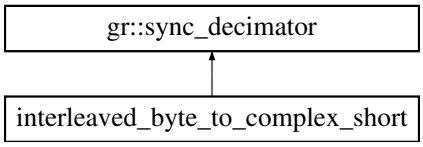
12.223.1 Detailed Description

This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::complex<unsigned char>).
Definition at line 40 of file interleaved_byte_to_complex_byte.h.
The documentation for this class was generated from the following file:

- interleaved_byte_to_complex_byte.h

12.224 interleaved_byte_to_complex_short Class Reference

This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.
`#include <interleaved_byte_to_complex_short.h>`
Inheritance diagram for interleaved_byte_to_complex_short:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- interleaved_byte_to_complex_short_sptr **make_interleaved_byte_to_complex_short** ()

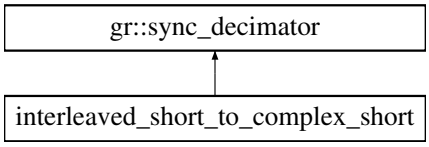
12.224.1 Detailed Description

This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.
Definition at line 40 of file interleaved_byte_to_complex_short.h.
The documentation for this class was generated from the following file:

- interleaved_byte_to_complex_short.h

12.225 interleaved_short_to_complex_short Class Reference

This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.
`#include <interleaved_short_to_complex_short.h>`
Inheritance diagram for interleaved_short_to_complex_short:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- interleaved_short_to_complex_short_sptr **make_interleaved_short_to_complex_short** ()

12.225.1 Detailed Description

This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.
Definition at line 39 of file interleaved_short_to_complex_short.h.
The documentation for this class was generated from the following file:

- interleaved_short_to_complex_short.h

12.226 IONGSMSChunkData Class Reference

Public Member Functions

- **IONGSMSChunkData** (const GnssMetadata::Chunk &chunk, const std::vector< std::string > &stream_ids, std::size_t output_stream_offset)
- **IONGSMSChunkData** (const IONGSMSChunkData &rhl)=delete
- IONGSMSChunkData & **operator=** (const IONGSMSChunkData &rhl)=delete
- **IONGSMSChunkData** (IONGSMSChunkData &&rhl)=delete
- IONGSMSChunkData & **operator=** (IONGSMSChunkData &&rhl)=delete
- std::size_t **read_from_buffer** (uint8_t *buffer, std::size_t offset)
- void **write_to_output** (gr_vector_void_star &outputs, std::vector< int > &output_items)
- std::size_t **output_stream_count** () const
- std::size_t **output_stream_item_size** (std::size_t stream_index) const
- std::size_t **output_stream_item_rate** (std::size_t stream_index) const

Static Public Member Functions

- static bool **stream_is_complex** (GnssMetadata::IonStream::SampleFormat format)
- static uint8_t **stream_output_item_bits** (const GnssMetadata::IonStream &stream)
- static std::size_t **stream_output_item_size** (const GnssMetadata::IonStream &stream)
- static std::size_t **stream_output_item_rate** (const GnssMetadata::IonStream &stream)

12.226.1 Detailed Description

Definition at line 112 of file ion_gsms_chunk_data.h.
The documentation for this class was generated from the following file:

- ion_gsms_chunk_data.h

12.227 IONGSMSChunkUnpackingCtx< WT > Struct Template Reference

Public Member Functions

- IONGSMSChunkUnpackingCtx (WT *data_buffer, std::size_t data_buffer_word_count, bool read_lsb_first)
- void shift_padding (std::size_t n_bits)
- uint64_t read_bits (std::size_t bit_count)

Public Attributes

- const WT * data_ = nullptr
- std::size_t word_count_ = 0
- std::size_t bit_offset_ = 0
- bool read_lsb_first_ = false

Static Public Attributes

- static constexpr uint8_t word_bitsize_ = sizeof(WT) * 8

12.227.1 Detailed Description

template<typename WT>
struct IONGSMSChunkUnpackingCtx< WT >

Definition at line 32 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.2 Constructor & Destructor Documentation

12.227.2.1 IONGSMSChunkUnpackingCtx()

```
template<typename WT>
IONGSMSChunkUnpackingCtx< WT >::IONGSMSChunkUnpackingCtx (
            WT * data_buffer,
            std::size_t data_buffer_word_count,
            bool read_lsb_first)  [inline]
```

Definition at line 41 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.3 Member Function Documentation

12.227.3.1 read_bits()

```
template<typename WT>
uint64_t IONGSMSChunkUnpackingCtx< WT >::read_bits (
            std::size_t bit_count)  [inline]
```

Definition at line 55 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.3.2 shift_padding()

```
template<typename WT>
void IONGSMSChunkUnpackingCtx< WT >::shift_padding (
            std::size_t n_bits)  [inline]
```

Definition at line 50 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.4 Member Data Documentation

12.227.4.1 bit_offset_

```
template<typename WT>
std::size_t IONGSMSChunkUnpackingCtx< WT >::bit_offset_ = 0
```

Definition at line 38 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.4.2 data_

```
template<typename WT>
const WT* IONGSMSChunkUnpackingCtx< WT >::data_ = nullptr
```

Definition at line 36 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.4.3 read_lsb_first_

```
template<typename WT>
bool IONGSMSChunkUnpackingCtx< WT >::read_lsb_first_ = false
```

Definition at line 39 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.4.4 word_bitsize_

```
template<typename WT>
uint8_t IONGSMSChunkUnpackingCtx< WT >::word_bitsize_ = sizeof(WT) * 8  [static], [constexpr]
```

Definition at line 34 of file ion_gsms_chunk_unpacking_ctx.h.

12.227.4.5 word_count_

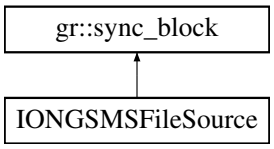
```
template<typename WT>
std::size_t IONGSMSChunkUnpackingCtx< WT >::word_count_ = 0
```

Definition at line 37 of file ion_gsms_chunk_unpacking_ctx.h.
The documentation for this struct was generated from the following file:

- ion_gsms_chunk_unpacking_ctx.h

12.228 IONGSMSFileSource Class Reference

Inheritance diagram for IONGSMSFileSource:



Classes

- struct SegmentDescriptor

Public Types

- using sptr = gnss_shared_ptr<IONGSMSFileSource>

Public Member Functions

- **IONGSMSFileSource** (const fs::path &metadata_filepath, const GnssMetadata::File &file, const Gnss↩ Metadata::Block &block, std::size_t block_start_offset, const std::vector< std::string > &stream_ids)
- **IONGSMSFileSource** (const std::vector< SegmentDescriptor > &segments, const std::vector< std::string > &stream_ids)
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override
- std::size_t **output_stream_count** () const
- std::size_t **output_stream_item_size** (std::size_t stream_index) const
- std::size_t **output_stream_total_sample_count** (std::size_t stream_index) const

12.228.1 Detailed Description

Definition at line 37 of file ion_gsms.h.

12.228.2 Member Typedef Documentation

12.228.2.1 sptr

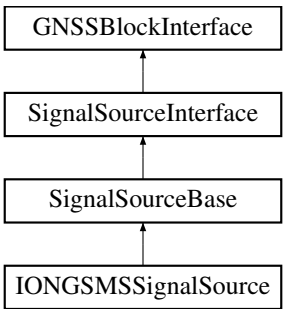
```
using IONGSMSFileSource::sptr = gnss_shared_ptr<IONGSMSFileSource>
```

Definition at line 40 of file ion_gsms.h.
The documentation for this class was generated from the following file:

- ion_gsms.h

12.229 IONGSMSSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <ion_gsms_signal_source.h>`
Inheritance diagram for IONGSMSSignalSource:



Public Member Functions

- **IONGSMSSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override
- size_t getRfChannels () const override
- size_t item_size () override

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.229.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 43 of file ion_gsms_signal_source.h.

12.229.2 Member Function Documentation

12.229.2.1 connect()

```
void IONGSMSSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Implements GNSSBlockInterface.

12.229.2.2 disconnect()

```
void IONGSMSSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Implements GNSSBlockInterface.

12.229.2.3 get_left_block()

```
gr::basic_block_sptr IONGSMSSignalSource::get_left_block ()  [override], [protected], [virtual]
```

Implements GNSSBlockInterface.

12.229.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr IONGSMSSignalSource::get_right_block ()  [override], [protected], [virtual]
```

Implements GNSSBlockInterface.

12.229.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr IONGSMSSignalSource::get_right_block (
            int RF_channel)  [override], [protected], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.229.2.6 getRfChannels()

```
size_t IONGSMSSignalSource::getRfChannels () const  [override], [protected], [virtual]
```

Implements SignalSourceInterface.

12.229.2.7 item_size()

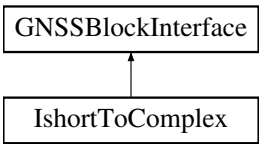
```
size_t IONGSMSSignalSource::item_size ()  [inline], [override], [protected], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 61 of file ion_gsms_signal_source.h.
The documentation for this class was generated from the following file:

- ion_gsms_signal_source.h

12.230 IshortToComplex Class Reference

Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.
`#include <ishort_to_complex.h>`
Inheritance diagram for IshortToComplex:



Public Member Functions

- **IshortToComplex** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Ishort_To_Complex".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.230.1 Detailed Description

Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.
Definition at line 39 of file ishort_to_complex.h.

12.230.2 Member Function Documentation

12.230.2.1 connect()

```
void IshortToComplex::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.230.2.2 disconnect()

```
void IshortToComplex::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.230.2.3 get_left_block()

```
gr::basic_block_sptr IshortToComplex::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.230.2.4 get_right_block()

```
gr::basic_block_sptr IshortToComplex::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.230.2.5 implementation()

```
std::string IshortToComplex::implementation ()  [inline], [override], [virtual]
```
Returns "Ishort_To_Complex".
Implements GNSSBlockInterface.
Definition at line 54 of file ishort_to_complex.h.

12.230.2.6 item_size()

```
size_t IshortToComplex::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 59 of file ishort_to_complex.h.

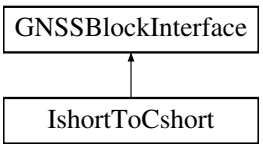
12.230.2.7 role()

```
std::string IshortToComplex::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 48 of file ishort_to_complex.h.
The documentation for this class was generated from the following file:

- ishort_to_complex.h

12.231 IshortToCshort Class Reference

Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.
`#include <ishort_to_cshort.h>`
Inheritance diagram for IshortToCshort:



Public Member Functions

- **IshortToCshort** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Ishort_To_Cshort".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.231.1 Detailed Description

Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.
Definition at line 39 of file ishort_to_cshort.h.

12.231.2 Member Function Documentation

12.231.2.1 connect()

```
void IshortToCshort::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.231.2.2 disconnect()

```
void IshortToCshort::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.231.2.3 get_left_block()

```
gr::basic_block_sptr IshortToCshort::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.231.2.4 get_right_block()

```
gr::basic_block_sptr IshortToCshort::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.231.2.5 implementation()

```
std::string IshortToCshort::implementation ()  [inline], [override], [virtual]
```

Returns "Ishort_To_Cshort".
Implements GNSSBlockInterface.
Definition at line 54 of file ishort_to_cshort.h.

12.231.2.6 item_size()

```
size_t IshortToCshort::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 59 of file ishort_to_cshort.h.

12.231.2.7 role()

```
std::string IshortToCshort::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 48 of file ishort_to_cshort.h.
The documentation for this class was generated from the following file:

- ishort_to_cshort.h

12.232 kernel_info_t Struct Reference

Public Attributes

- cl_kernel kernel
- char * kernel_name
- unsigned lmem_size
- unsigned num_workgroups
- unsigned num_xforms_per_workgroup

- unsigned num_workitems_per_workgroup
- cl_fft_kernel_dir dir
- int in_place_possible
- kernel_info_t * next

12.232.1 Detailed Description

Definition at line 32 of file fft_internal.h.

12.232.2 Member Data Documentation

12.232.2.1 dir

```
cl_fft_kernel_dir kernel_info_t::dir
```

Definition at line 40 of file fft_internal.h.

12.232.2.2 in_place_possible

```
int kernel_info_t::in_place_possible
```

Definition at line 41 of file fft_internal.h.

12.232.2.3 kernel

```
cl_kernel kernel_info_t::kernel
```

Definition at line 34 of file fft_internal.h.

12.232.2.4 kernel_name

```
char* kernel_info_t::kernel_name
```

Definition at line 35 of file fft_internal.h.

12.232.2.5 lmem_size

```
unsigned kernel_info_t::lmem_size
```

Definition at line 36 of file fft_internal.h.

12.232.2.6 next

```
kernel_info_t* kernel_info_t::next
```

Definition at line 42 of file fft_internal.h.

12.232.2.7 num_workgroups

```
unsigned kernel_info_t::num_workgroups
```

Definition at line 37 of file fft_internal.h.

12.232.2.8 num_workitems_per_workgroup

```
unsigned kernel_info_t::num_workitems_per_workgroup
```

Definition at line 39 of file fft_internal.h.

12.232.2.9 num_xforms_per_workgroup

```
unsigned kernel_info_t::num_xforms_per_workgroup
```

Definition at line 38 of file fft_internal.h.

The documentation for this struct was generated from the following file:

- fft_internal.h

12.233 Kf_Conf Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role)

Public Attributes

- std::string item_type
- std::string dump_filename
- double fs_in
- double carrier_lock_th
- double code_disc_sd_chips
- double carrier_disc_sd_rads
- double code_phase_sd_chips
- double carrier_phase_sd_rad
- double carrier_freq_sd_hz
- double carrier_freq_rate_sd_hz_s
- double init_code_phase_sd_chips
- double init_carrier_phase_sd_rad
- double init_carrier_freq_sd_hz
- double init_carrier_freq_rate_sd_hz_s
- float early_late_space_chips
- float very_early_late_space_chips
- float early_late_space_narrow_chips
- float very_early_late_space_narrow_chips
- float slope
- float spc
- float y_intercept
- float cn0_smoother_alpha
- float carrier_lock_test_smoother_alpha
- uint32_t pull_in_time_s
- uint32_t bit_synchronization_time_limit_s
- uint32_t vector_length
- uint32_t smoother_length
- int32_t extend_correlation_symbols
- int32_t cn0_samples
- int32_t cn0_smoother_samples
- int32_t carrier_lock_test_smoother_samples
- int32_t cn0_min
- int32_t max_code_lock_fail
- int32_t max_carrier_lock_fail
- char signal [3] {}
- char system
- bool track_pilot
- bool enable_doppler_correction
- bool high_dyn
- bool dump
- bool dump_mat

12.233.1 Detailed Description

Definition at line 29 of file kf_conf.h.

12.233.2 Member Data Documentation

12.233.2.1 bit_synchronization_time_limit_s

```
uint32_t Kf_Conf::bit_synchronization_time_limit_s
```
Definition at line 67 of file kf_conf.h.

12.233.2.2 carrier_disc_sd_rads

```
double Kf_Conf::carrier_disc_sd_rads
```
Definition at line 43 of file kf_conf.h.

12.233.2.3 carrier_freq_rate_sd_hz_s

```
double Kf_Conf::carrier_freq_rate_sd_hz_s
```
Definition at line 49 of file kf_conf.h.

12.233.2.4 carrier_freq_sd_hz

```
double Kf_Conf::carrier_freq_sd_hz
```
Definition at line 48 of file kf_conf.h.

12.233.2.5 carrier_lock_test_smoother_alpha

```
float Kf_Conf::carrier_lock_test_smoother_alpha
```
Definition at line 65 of file kf_conf.h.

12.233.2.6 carrier_lock_test_smoother_samples

```
int32_t Kf_Conf::carrier_lock_test_smoother_samples
```
Definition at line 73 of file kf_conf.h.

12.233.2.7 carrier_lock_th

```
double Kf_Conf::carrier_lock_th
```
Definition at line 38 of file kf_conf.h.

12.233.2.8 carrier_phase_sd_rad

```
double Kf_Conf::carrier_phase_sd_rad
```
Definition at line 47 of file kf_conf.h.

12.233.2.9 cn0_min

```
int32_t Kf_Conf::cn0_min
```
Definition at line 74 of file kf_conf.h.

12.233.2.10 cn0_samples

```
int32_t Kf_Conf::cn0_samples
```
Definition at line 71 of file kf_conf.h.

12.233.2.11 cn0_smoother_alpha

```
float Kf_Conf::cn0_smoother_alpha
```
Definition at line 64 of file kf_conf.h.

12.233.2.12 cn0_smoother_samples

```
int32_t Kf_Conf::cn0_smoother_samples
```
Definition at line 72 of file kf_conf.h.

12.233.2.13 code_disc_sd_chips

`double Kf_Conf::code_disc_sd_chips`
Definition at line 42 of file kf_conf.h.

12.233.2.14 code_phase_sd_chips

`double Kf_Conf::code_phase_sd_chips`
Definition at line 46 of file kf_conf.h.

12.233.2.15 dump

`bool Kf_Conf::dump`
Definition at line 82 of file kf_conf.h.

12.233.2.16 dump_filename

`std::string Kf_Conf::dump_filename`
Definition at line 36 of file kf_conf.h.

12.233.2.17 dump_mat

`bool Kf_Conf::dump_mat`
Definition at line 83 of file kf_conf.h.

12.233.2.18 early_late_space_chips

`float Kf_Conf::early_late_space_chips`
Definition at line 57 of file kf_conf.h.

12.233.2.19 early_late_space_narrow_chips

`float Kf_Conf::early_late_space_narrow_chips`
Definition at line 59 of file kf_conf.h.

12.233.2.20 enable_doppler_correction

`bool Kf_Conf::enable_doppler_correction`
Definition at line 80 of file kf_conf.h.

12.233.2.21 extend_correlation_symbols

`int32_t Kf_Conf::extend_correlation_symbols`
Definition at line 70 of file kf_conf.h.

12.233.2.22 fs_in

`double Kf_Conf::fs_in`
Definition at line 37 of file kf_conf.h.

12.233.2.23 high_dyn

`bool Kf_Conf::high_dyn`
Definition at line 81 of file kf_conf.h.

12.233.2.24 init_carrier_freq_rate_sd_hz_s

`double Kf_Conf::init_carrier_freq_rate_sd_hz_s`
Definition at line 55 of file kf_conf.h.

12.233.2.25 init_carrier_freq_sd_hz

```
double Kf_Conf::init_carrier_freq_sd_hz
```
Definition at line 54 of file kf_conf.h.

12.233.2.26 init_carrier_phase_sd_rad

```
double Kf_Conf::init_carrier_phase_sd_rad
```
Definition at line 53 of file kf_conf.h.

12.233.2.27 init_code_phase_sd_chips

```
double Kf_Conf::init_code_phase_sd_chips
```
Definition at line 52 of file kf_conf.h.

12.233.2.28 item_type

```
std::string Kf_Conf::item_type
```
Definition at line 35 of file kf_conf.h.

12.233.2.29 max_carrier_lock_fail

```
int32_t Kf_Conf::max_carrier_lock_fail
```
Definition at line 76 of file kf_conf.h.

12.233.2.30 max_code_lock_fail

```
int32_t Kf_Conf::max_code_lock_fail
```
Definition at line 75 of file kf_conf.h.

12.233.2.31 pull_in_time_s

```
uint32_t Kf_Conf::pull_in_time_s
```
Definition at line 66 of file kf_conf.h.

12.233.2.32 signal

```
char Kf_Conf::signal[3] {}
```
Definition at line 77 of file kf_conf.h.

12.233.2.33 slope

```
float Kf_Conf::slope
```
Definition at line 61 of file kf_conf.h.

12.233.2.34 smoother_length

```
uint32_t Kf_Conf::smoother_length
```
Definition at line 69 of file kf_conf.h.

12.233.2.35 spc

```
float Kf_Conf::spc
```
Definition at line 62 of file kf_conf.h.

12.233.2.36 system

```
char Kf_Conf::system
```
Definition at line 78 of file kf_conf.h.

12.233.2.37 track_pilot

`bool Kf_Conf::track_pilot`
Definition at line 79 of file kf_conf.h.

12.233.2.38 vector_length

`uint32_t Kf_Conf::vector_length`
Definition at line 68 of file kf_conf.h.

12.233.2.39 very_early_late_space_chips

`float Kf_Conf::very_early_late_space_chips`
Definition at line 58 of file kf_conf.h.

12.233.2.40 very_early_late_space_narrow_chips

`float Kf_Conf::very_early_late_space_narrow_chips`
Definition at line 60 of file kf_conf.h.

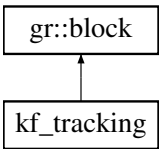
12.233.2.41 y_intercept

`float Kf_Conf::y_intercept`
Definition at line 63 of file kf_conf.h.
The documentation for this class was generated from the following file:

- kf_conf.h

12.234 kf_tracking Class Reference

This class implements a code DLL + carrier PLL tracking block.
`#include <kf_tracking.h>`
Inheritance diagram for kf_tracking:



Public Member Functions

- void **set_channel** (uint32_t channel)
- void **set_gnss_synchro** (Gnss_Synchro *p_gnss_synchro)
- void **start_tracking** ()
- void **stop_tracking** ()
- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)

Friends

- kf_tracking_sptr **kf_make_tracking** (const Kf_Conf &conf_)

12.234.1 Detailed Description

This class implements a code DLL + carrier PLL tracking block.
Definition at line 60 of file kf_tracking.h.
The documentation for this class was generated from the following file:

- kf_tracking.h

12.235 Kml_Printer Class Reference

Prints PVT information to OGC KML format file (can be viewed with Google Earth).
`#include <kml_printer.h>`

Public Member Functions

- **Kml_Printer** (const std::string &base_path=std::string("."))
- bool **set_headers** (const std::string &filename, bool time_tag_name=true)
- bool **print_position** (const Pvt_Solution *const position)
- bool **close_file** ()

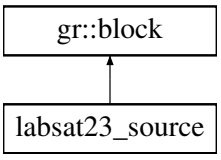
12.235.1 Detailed Description

Prints PVT information to OGC KML format file (can be viewed with Google Earth).
See https://www.opengeospatial.org/standards/kml
Definition at line 38 of file kml_printer.h.
The documentation for this class was generated from the following file:

- kml_printer.h

12.236 labsat23_source Class Reference

This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.
`#include <labsat23_source.h>`
Inheritance diagram for labsat23_source:



Public Member Functions

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- labsat23_source_sptr **labsat23_make_source_sptr** (const char *signal_file_basename, const std::vector< int > &channel_selector, Concurrent_Queue< pmt::pmt_t > *queue, bool digital_io_enabled, double seconds_to_skip)

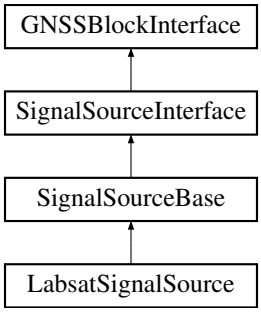
12.236.1 Detailed Description

This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.
Definition at line 57 of file labsat23_source.h.
The documentation for this class was generated from the following file:

- labsat23_source.h

12.237 LabsatSignalSource Class Reference

This class reads samples stored in LabSat version 2, 3, and 3 Wideband format.
`#include <labsat_signal_source.h>`
Inheritance diagram for LabsatSignalSource:



Public Member Functions

- **LabsatSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- size_t getRfChannels () const override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int i) override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.237.1 Detailed Description

This class reads samples stored in LabSat version 2, 3, and 3 Wideband format.
Definition at line 44 of file labsat_signal_source.h.

12.237.2 Member Function Documentation

12.237.2.1 connect()

```
void LabsatSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.237.2.2 disconnect()

```
void LabsatSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.237.2.3 get_left_block()

```
gr::basic_block_sptr LabsatSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.237.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr LabsatSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.237.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr LabsatSignalSource::get_right_block (
            int i)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.237.2.6 getRfChannels()

```
size_t LabsatSignalSource::getRfChannels () const  [override], [virtual]
```

Implements SignalSourceInterface.

12.237.2.7 item_size()

```
size_t LabsatSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 53 of file labsat_signal_source.h.
The documentation for this class was generated from the following file:

- labsat_signal_source.h

12.238 lex_t Struct Reference

Public Attributes

- int n
- int nmax
- lexmsg_t * msgs

12.238.1 Detailed Description

Definition at line 724 of file rtklib.h.

12.238.2 Member Data Documentation

12.238.2.1 msgs

```
lexmsg_t* lex_t::msgs
```
Definition at line 727 of file rtklib.h.

12.238.2.2 n

```
int lex_t::n
```
Definition at line 726 of file rtklib.h.

12.238.2.3 nmax

```
int lex_t::nmax
```
Definition at line 726 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.239 lexeph_t Struct Reference

Public Attributes

- gtime_t toe
- gtime_t tof
- int sat
- unsigned char health
- unsigned char ura
- double pos [3]
- double vel [3]
- double acc [3]
- double jerk [3]
- double af0
- double af1
- double tgd
- double isc [8]

12.239.1 Detailed Description

Definition at line 731 of file rtklib.h.

12.239.2 Member Data Documentation

12.239.2.1 acc

```
double lexeph_t::acc[3]
```
Definition at line 740 of file rtklib.h.

12.239.2.2 af0

```
double lexeph_t::af0
```
Definition at line 742 of file rtklib.h.

12.239.2.3 af1

```
double lexeph_t::af1
```
Definition at line 742 of file rtklib.h.

12.239.2.4 health

```
unsigned char lexeph_t::health
```
Definition at line 736 of file rtklib.h.

12.239.2.5 isc

```
double lexeph_t::isc[8]
```
Definition at line 744 of file rtklib.h.

12.239.2.6 jerk

```
double lexeph_t::jerk[3]
```
Definition at line 741 of file rtklib.h.

12.239.2.7 pos

```
double lexeph_t::pos[3]
```
Definition at line 738 of file rtklib.h.

12.239.2.8 sat

```
int lexeph_t::sat
```
Definition at line 735 of file rtklib.h.

12.239.2.9 tgd

```
double lexeph_t::tgd
```
Definition at line 743 of file rtklib.h.

12.239.2.10 toe

```
gtime_t lexeph_t::toe
```
Definition at line 733 of file rtklib.h.

12.239.2.11 tof

```
gtime_t lexeph_t::tof
```
Definition at line 734 of file rtklib.h.

12.239.2.12 ura

```
unsigned char lexeph_t::ura
```
Definition at line 737 of file rtklib.h.

12.239.2.13 vel

```
double lexeph_t::vel[3]
```
Definition at line 739 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.240 lexion_t Struct Reference

Public Attributes

- gtime_t t0
- double tspan
- double pos0 [2]
- double coef [3][2]

12.240.1 Detailed Description

Definition at line 748 of file rtklib.h.

12.240.2 Member Data Documentation

12.240.2.1 coef

```
double lexion_t::coef[3][2]
```
Definition at line 753 of file rtklib.h.

12.240.2.2 pos0

```
double lexion_t::pos0[2]
```
Definition at line 752 of file rtklib.h.

12.240.2.3 t0

```
gtime_t lexion_t::t0
```
Definition at line 750 of file rtklib.h.

12.240.2.4 tspan

```
double lexion_t::tspan
```
Definition at line 751 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.241 lexmsg_t Struct Reference

Public Attributes

- int prn
- int type
- int alert
- unsigned char stat
- unsigned char snr
- unsigned int ttt
- unsigned char msg [212]

12.241.1 Detailed Description

Definition at line 712 of file rtklib.h.

12.241.2 Member Data Documentation

12.241.2.1 alert

```
int lexmsg_t::alert
```
Definition at line 716 of file rtklib.h.

12.241.2.2 msg

```
unsigned char lexmsg_t::msg[212]
```
Definition at line 720 of file rtklib.h.

12.241.2.3 prn

```
int lexmsg_t::prn
```
Definition at line 714 of file rtklib.h.

12.241.2.4 snr

`unsigned char lexmsg_t::snr`
Definition at line 718 of file rtklib.h.

12.241.2.5 stat

`unsigned char lexmsg_t::stat`
Definition at line 717 of file rtklib.h.

12.241.2.6 ttt

`unsigned int lexmsg_t::ttt`
Definition at line 719 of file rtklib.h.

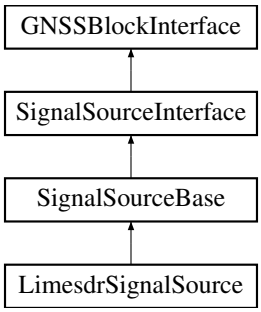
12.241.2.7 type

`int lexmsg_t::type`
Definition at line 715 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.242 LimesdrSignalSource Class Reference

This class instantiates the LimeSDR gnuradio signal source. It has support also for a customized LimeSDR firmware and signal source to support PPS samplestamp reading.
`#include <limesdr_signal_source.h>`
Inheritance diagram for LimesdrSignalSource:



Public Member Functions

- **LimesdrSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in↩ _stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.242.1 Detailed Description

This class instantiates the LimeSDR gnuradio signal source. It has support also for a customized LimeSDR firmware and signal source to support PPS samplestamp reading.
Definition at line 42 of file limesdr_signal_source.h.

12.242.2 Member Function Documentation

12.242.2.1 connect()

```
void LimesdrSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.242.2.2 disconnect()

```
void LimesdrSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.242.2.3 get_left_block()

```
gr::basic_block_sptr LimesdrSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.242.2.4 get_right_block()

```
gr::basic_block_sptr LimesdrSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.242.2.5 item_size()

```
size_t LimesdrSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 51 of file limesdr_signal_source.h.
The documentation for this class was generated from the following file:

- limesdr_signal_source.h

12.243 MACK_header Class Reference

Public Attributes

- uint64_t tag0 {}
- uint16_t macseq {}
- uint8_t cop {}
- bool tag0_valid {false}
- bool macseq_valid {false}

12.243.1 Detailed Description

Definition at line 68 of file osnma_data.h.

12.243.2 Member Data Documentation

12.243.2.1 cop

```
uint8_t MACK_header::cop {}
```

Definition at line 74 of file osnma_data.h.

12.243.2.2 macseq

```
uint16_t MACK_header::macseq {}
```

Definition at line 73 of file osnma_data.h.

12.243.2.3 macseq_valid

```
bool MACK_header::macseq_valid {false}
```

Definition at line 76 of file osnma_data.h.

12.243.2.4 tag0

```
uint64_t MACK_header::tag0 {}
```

Definition at line 72 of file osnma_data.h.

12.243.2.5 tag0_valid

```
bool MACK_header::tag0_valid {false}
```

Definition at line 75 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.244 Mack_lookup Class Reference

Public Member Functions

- Mack_lookup (uint8_t msg_, uint8_t nt_, const std::vector< std::string > &s1_, const std::vector< std::string > &s2_)

Public Attributes

- uint8_t msg {}
- uint8_t nt {}
- std::vector< std::string > sequence1
- std::vector< std::string > sequence2

12.244.1 Detailed Description

Definition at line 166 of file Galileo_OSNMA.h.

12.244.2 Constructor & Destructor Documentation

12.244.2.1 Mack_lookup()

```
Mack_lookup::Mack_lookup (
            uint8_t msg_,
            uint8_t nt_,
            const std::vector< std::string > & s1_,
            const std::vector< std::string > & s2_)  [inline]
```

Definition at line 170 of file Galileo_OSNMA.h.

12.244.3 Member Data Documentation

12.244.3.1 msg

```
uint8_t Mack_lookup::msg {}
```

Definition at line 177 of file Galileo_OSNMA.h.

12.244.3.2 nt

```
uint8_t Mack_lookup::nt {}
```

Definition at line 178 of file Galileo_OSNMA.h.

12.244.3.3 sequence1

```
std::vector<std::string> Mack_lookup::sequence1
```

Definition at line 179 of file Galileo_OSNMA.h.

12.244.3.4 sequence2

```
std::vector<std::string> Mack_lookup::sequence2
```

Definition at line 180 of file Galileo_OSNMA.h.

The documentation for this class was generated from the following file:

- Galileo_OSNMA.h

12.245 MACK_message Class Reference

Public Attributes

- MACK_header header
- std::vector< MACK_tag_and_info > tag_and_info
- std::vector< uint8_t > key
- std::vector< uint8_t > allowed_adkds {0, 4, 12}
- uint32_t TOW {}
- uint32_t WN {}
- uint32_t PRNa {}
- uint8_t nmas {}

12.245.1 Detailed Description

Definition at line 141 of file osnma_data.h.

12.245.2 Member Data Documentation

12.245.2.1 allowed_adkds

```
std::vector<uint8_t> MACK_message::allowed_adkds {0, 4, 12}
```
Definition at line 148 of file osnma_data.h.

12.245.2.2 header

```
MACK_header MACK_message::header
```
Definition at line 145 of file osnma_data.h.

12.245.2.3 key

```
std::vector<uint8_t> MACK_message::key
```
Definition at line 147 of file osnma_data.h.

12.245.2.4 nmas

```
uint8_t MACK_message::nmas {}
```
Definition at line 152 of file osnma_data.h.

12.245.2.5 PRNa

```
uint32_t MACK_message::PRNa {}
```
Definition at line 151 of file osnma_data.h.

12.245.2.6 tag_and_info

```
std::vector<MACK_tag_and_info> MACK_message::tag_and_info
```
Definition at line 146 of file osnma_data.h.

12.245.2.7 TOW

```
uint32_t MACK_message::TOW {}
```
Definition at line 149 of file osnma_data.h.

12.245.2.8 WN

```
uint32_t MACK_message::WN {}
```
Definition at line 150 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.246 MACK_tag_and_info Class Reference

Public Attributes

- uint64_t tag {}
- MACK_tag_info tag_info
- uint32_t counter {}
- bool valid {true}

12.246.1 Detailed Description

Definition at line 90 of file osnma_data.h.

12.246.2 Member Data Documentation

12.246.2.1 counter

```
uint32_t MACK_tag_and_info::counter {}
```
Definition at line 96 of file osnma_data.h.

12.246.2.2 tag

```
uint64_t MACK_tag_and_info::tag {}
```
Definition at line 94 of file osnma_data.h.

12.246.2.3 tag_info

```
MACK_tag_info MACK_tag_and_info::tag_info
```
Definition at line 95 of file osnma_data.h.

12.246.2.4 valid

```
bool MACK_tag_and_info::valid {true}
```
Definition at line 97 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.247 MACK_tag_info Class Reference

Public Attributes

- uint8_t PRN_d {}
- uint8_t ADKD {}
- uint8_t cop {}

12.247.1 Detailed Description

Definition at line 80 of file osnma_data.h.

12.247.2 Member Data Documentation

12.247.2.1 ADKD

```
uint8_t MACK_tag_info::ADKD {}
```
Definition at line 85 of file osnma_data.h.

12.247.2.2 cop

```
uint8_t MACK_tag_info::cop {}
```
Definition at line 86 of file osnma_data.h.

12.247.2.3 PRN_d

```
uint8_t MACK_tag_info::PRN_d {}
```
Definition at line 84 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.248 matlab_type_traits< T > Struct Template Reference

12.248.1 Detailed Description

template<typename T>
struct matlab_type_traits< T >

Definition at line 26 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.249 matlab_type_traits< double > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_DOUBLE
- static constexpr matio_types data_type = MAT_T_DOUBLE

12.249.1 Detailed Description

Definition at line 71 of file matlab_writter_helper.h.

12.249.2 Member Data Documentation

12.249.2.1 class_type

```
matio_classes matlab_type_traits< double >::class_type = MAT_C_DOUBLE  [static], [constexpr]
```

Definition at line 73 of file matlab_writter_helper.h.

12.249.2.2 data_type

```
matio_types matlab_type_traits< double >::data_type = MAT_T_DOUBLE  [static], [constexpr]
```

Definition at line 74 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.250 matlab_type_traits< float > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_SINGLE
- static constexpr matio_types data_type = MAT_T_SINGLE

12.250.1 Detailed Description

Definition at line 64 of file matlab_writter_helper.h.

12.250.2 Member Data Documentation

12.250.2.1 class_type

```
matio_classes matlab_type_traits< float >::class_type = MAT_C_SINGLE  [static], [constexpr]
```

Definition at line 66 of file matlab_writter_helper.h.

12.250.2.2 data_type

```
matio_types matlab_type_traits< float >::data_type = MAT_T_SINGLE  [static], [constexpr]
```

Definition at line 67 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.251 matlab_type_traits< int32_t > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_INT32
- static constexpr matio_types data_type = MAT_T_INT32

12.251.1 Detailed Description

Definition at line 29 of file matlab_writter_helper.h.

12.251.2 Member Data Documentation

12.251.2.1 class_type

```
matio_classes matlab_type_traits< int32_t >::class_type = MAT_C_INT32  [static], [constexpr]
```

Definition at line 31 of file matlab_writter_helper.h.

12.251.2.2 data_type

```
matio_types matlab_type_traits< int32_t >::data_type = MAT_T_INT32  [static], [constexpr]
```

Definition at line 32 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.252 matlab_type_traits< int64_t > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_INT64
- static constexpr matio_types data_type = MAT_T_INT64

12.252.1 Detailed Description

Definition at line 50 of file matlab_writter_helper.h.

12.252.2 Member Data Documentation

12.252.2.1 class_type

```
matio_classes matlab_type_traits< int64_t >::class_type = MAT_C_INT64  [static], [constexpr]
```

Definition at line 52 of file matlab_writter_helper.h.

12.252.2.2 data_type

```
matio_types matlab_type_traits< int64_t >::data_type = MAT_T_INT64  [static], [constexpr]
```

Definition at line 53 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.253 matlab_type_traits< uint32_t > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_UINT32
- static constexpr matio_types data_type = MAT_T_UINT32

12.253.1 Detailed Description

Definition at line 36 of file matlab_writter_helper.h.

12.253.2 Member Data Documentation

12.253.2.1 class_type

```
matio_classes matlab_type_traits< uint32_t >::class_type = MAT_C_UINT32  [static], [constexpr]
```

Definition at line 38 of file matlab_writter_helper.h.

12.253.2.2 data_type

```
matio_types matlab_type_traits< uint32_t >::data_type = MAT_T_UINT32  [static], [constexpr]
```

Definition at line 39 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.254 matlab_type_traits< uint64_t > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_UINT64
- static constexpr matio_types data_type = MAT_T_UINT64

12.254.1 Detailed Description

Definition at line 57 of file matlab_writter_helper.h.

12.254.2 Member Data Documentation

12.254.2.1 class_type

```
matio_classes matlab_type_traits< uint64_t >::class_type = MAT_C_UINT64  [static], [constexpr]
```

Definition at line 59 of file matlab_writter_helper.h.

12.254.2.2 data_type

```
matio_types matlab_type_traits< uint64_t >::data_type = MAT_T_UINT64  [static], [constexpr]
```

Definition at line 60 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.255 matlab_type_traits< uint8_t > Struct Reference

Static Public Attributes

- static constexpr matio_classes class_type = MAT_C_UINT8
- static constexpr matio_types data_type = MAT_T_UINT8

12.255.1 Detailed Description

Definition at line 43 of file matlab_writter_helper.h.

12.255.2 Member Data Documentation

12.255.2.1 class_type

```
matio_classes matlab_type_traits< uint8_t >::class_type = MAT_C_UINT8  [static], [constexpr]
```

Definition at line 45 of file matlab_writter_helper.h.

12.255.2.2 data_type

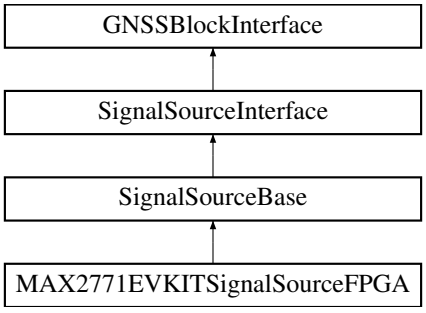
```
matio_types matlab_type_traits< uint8_t >::data_type = MAT_T_UINT8  [static], [constexpr]
```

Definition at line 46 of file matlab_writter_helper.h.
The documentation for this struct was generated from the following file:

- matlab_writter_helper.h

12.256 MAX2771EVKITSignalSourceFPGA Class Reference

Inheritance diagram for MAX2771EVKITSignalSourceFPGA:



Public Member Functions

- **MAX2771EVKITSignalSourceFPGA** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- std::vector< uint32_t > **setup_regs** (uint64_t freq)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.256.1 Detailed Description

Definition at line 45 of file max2771_evkit_signal_source_fpga.h.

12.256.2 Member Function Documentation

12.256.2.1 connect()

```
void MAX2771EVKITSignalSourceFPGA::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.256.2.2 disconnect()

```
void MAX2771EVKITSignalSourceFPGA::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.256.2.3 get_left_block()

```
gr::basic_block_sptr MAX2771EVKITSignalSourceFPGA::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.256.2.4 get_right_block()

```
gr::basic_block_sptr MAX2771EVKITSignalSourceFPGA::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.256.2.5 item_size()

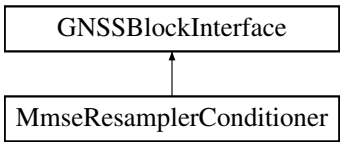
```
size_t MAX2771EVKITSignalSourceFPGA::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 56 of file max2771_evkit_signal_source_fpga.h.
The documentation for this class was generated from the following file:

- max2771_evkit_signal_source_fpga.h

12.257 MmseResamplerConditioner Class Reference

Interface of a MMSE resampler block adapter to a SignalConditionerInterface.
`#include <mmse_resampler_conditioner.h>`
Inheritance diagram for MmseResamplerConditioner:



Public Member Functions

- **MmseResamplerConditioner** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream)
- std::string role () override
- std::string implementation () override
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.257.1 Detailed Description

Interface of a MMSE resampler block adapter to a SignalConditionerInterface.
Definition at line 48 of file mmse_resampler_conditioner.h.

12.257.2 Member Function Documentation

12.257.2.1 connect()

```
void MmseResamplerConditioner::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.257.2.2 disconnect()

```
void MmseResamplerConditioner::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.257.2.3 get_left_block()

```
gr::basic_block_sptr MmseResamplerConditioner::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.257.2.4 get_right_block()

```
gr::basic_block_sptr MmseResamplerConditioner::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.257.2.5 implementation()

```
std::string MmseResamplerConditioner::implementation ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 62 of file mmse_resampler_conditioner.h.

12.257.2.6 item_size()

```
size_t MmseResamplerConditioner::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 67 of file mmse_resampler_conditioner.h.

12.257.2.7 role()

```
std::string MmseResamplerConditioner::role () [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 57 of file mmse_resampler_conditioner.h.
The documentation for this class was generated from the following file:

- mmse_resampler_conditioner.h

12.258 ModelFunction Class Reference

Public Member Functions

- virtual arma::vec **operator()** (const arma::vec &input)=0

12.258.1 Detailed Description

Definition at line 46 of file nonlinear_tracking.h.

12.258.2 Constructor & Destructor Documentation

12.258.2.1 ModelFunction()

```
ModelFunction::ModelFunction () [inline]
```

Definition at line 49 of file nonlinear_tracking.h.
The documentation for this class was generated from the following file:

- nonlinear_tracking.h

12.259 Monitor_Ephemeris_Udp_Sink Class Reference

Public Member Functions

- **Monitor_Ephemeris_Udp_Sink** (const std::vector< std::string > &addresses, const uint16_t &port, bool protobuf_enabled)
- bool **write_gps_ephemeris** (const std::shared_ptr< Gps_Ephemeris > &monitor_gps_eph)
- bool **write_galileo_ephemeris** (const std::shared_ptr< Galileo_Ephemeris > &monitor_gal_eph)

12.259.1 Detailed Description

Definition at line 42 of file monitor_ephemeris_udp_sink.h.
The documentation for this class was generated from the following file:

- monitor_ephemeris_udp_sink.h

12.260 Monitor_Pvt Class Reference

This class contains parameters and outputs of the PVT block.
`#include <monitor_pvt.h>`

Classes

- class TrackedSatelliteInfo

  One tracked satellite/signal with its azimuth/elevation, whether it was combined with another signal of the same satellite (e.g. Galileo E1+E5a iono-free combination) and whether it was used in this fix. Signals are listed individually (one entry per satellite per signal): a combined satellite appears as two entries, both with combined = true.

Public Member Functions

- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

  This member function serializes and restores Monitor_Pvt objects from a byte stream.

Public Attributes

- std::vector< TrackedSatelliteInfo > tracked_satellites
- uint32_t TOW_at_current_symbol_ms
- uint32_t week
- double RX_time
- double user_clk_offset
- double pos_x
- double pos_y
- double pos_z
- double vel_x
- double vel_y
- double vel_z
- double cov_xx
- double cov_yy
- double cov_zz
- double cov_xy
- double cov_yz
- double cov_zx
- double latitude
- double longitude
- double height
- double vel_e
- double vel_n
- double vel_u
- double cog
- uint32_t galhas_status
- uint8_t valid_sats
- uint8_t solution_status
- uint8_t solution_type
- float AR_ratio_factor
- float AR_ratio_threshold
- double gdop
- double pdop
- double hdop
- double vdop
- double user_clk_drift_ppm
- std::string utc_time
- std::string geohash

12.260.1 Detailed Description

This class contains parameters and outputs of the PVT block.
Definition at line 35 of file monitor_pvt.h.

12.260.2 Member Function Documentation

12.260.2.1 serialize()

```
template<class Archive>
void Monitor_Pvt::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

This member function serializes and restores Monitor_Pvt objects from a byte stream.
Definition at line 153 of file monitor_pvt.h.

12.260.3 Member Data Documentation

12.260.3.1 AR_ratio_factor

```
float Monitor_Pvt::AR_ratio_factor
```

Definition at line 129 of file monitor_pvt.h.

12.260.3.2 AR_ratio_threshold

```
float Monitor_Pvt::AR_ratio_threshold
```

Definition at line 131 of file monitor_pvt.h.

12.260.3.3 cog

```
double Monitor_Pvt::cog
```

Definition at line 117 of file monitor_pvt.h.

12.260.3.4 cov_xx

```
double Monitor_Pvt::cov_xx
```

Definition at line 98 of file monitor_pvt.h.

12.260.3.5 cov_xy

```
double Monitor_Pvt::cov_xy
```

Definition at line 101 of file monitor_pvt.h.

12.260.3.6 cov_yy

```
double Monitor_Pvt::cov_yy
```

Definition at line 99 of file monitor_pvt.h.

12.260.3.7 cov_yz

```
double Monitor_Pvt::cov_yz
```

Definition at line 102 of file monitor_pvt.h.

12.260.3.8 cov_zx

```
double Monitor_Pvt::cov_zx
```

Definition at line 103 of file monitor_pvt.h.

12.260.3.9 cov_zz

```
double Monitor_Pvt::cov_zz
```

Definition at line 100 of file monitor_pvt.h.

12.260.3.10 galhas_status

`uint32_t Monitor_Pvt::galhas_status`
Definition at line 120 of file monitor_pvt.h.

12.260.3.11 gdop

`double Monitor_Pvt::gdop`
Definition at line 134 of file monitor_pvt.h.

12.260.3.12 geohash

`std::string Monitor_Pvt::geohash`
Definition at line 145 of file monitor_pvt.h.

12.260.3.13 hdop

`double Monitor_Pvt::hdop`
Definition at line 136 of file monitor_pvt.h.

12.260.3.14 height

`double Monitor_Pvt::height`
Definition at line 110 of file monitor_pvt.h.

12.260.3.15 latitude

`double Monitor_Pvt::latitude`
Definition at line 106 of file monitor_pvt.h.

12.260.3.16 longitude

`double Monitor_Pvt::longitude`
Definition at line 108 of file monitor_pvt.h.

12.260.3.17 pdop

`double Monitor_Pvt::pdop`
Definition at line 135 of file monitor_pvt.h.

12.260.3.18 pos_x

`double Monitor_Pvt::pos_x`
Definition at line 90 of file monitor_pvt.h.

12.260.3.19 pos_y

`double Monitor_Pvt::pos_y`
Definition at line 91 of file monitor_pvt.h.

12.260.3.20 pos_z

`double Monitor_Pvt::pos_z`
Definition at line 92 of file monitor_pvt.h.

12.260.3.21 RX_time

`double Monitor_Pvt::RX_time`
Definition at line 85 of file monitor_pvt.h.

12.260.3.22 solution_status

`uint8_t Monitor_Pvt::solution_status`
Definition at line 125 of file monitor_pvt.h.

12.260.3.23 solution_type

`uint8_t Monitor_Pvt::solution_type`
Definition at line 127 of file monitor_pvt.h.

12.260.3.24 TOW_at_current_symbol_ms

`uint32_t Monitor_Pvt::TOW_at_current_symbol_ms`
Definition at line 81 of file monitor_pvt.h.

12.260.3.25 tracked_satellites

`std::vector<TrackedSatelliteInfo> Monitor_Pvt::tracked_satellites`
Definition at line 78 of file monitor_pvt.h.

12.260.3.26 user_clk_drift_ppm

`double Monitor_Pvt::user_clk_drift_ppm`
Definition at line 140 of file monitor_pvt.h.

12.260.3.27 user_clk_offset

`double Monitor_Pvt::user_clk_offset`
Definition at line 87 of file monitor_pvt.h.

12.260.3.28 utc_time

`std::string Monitor_Pvt::utc_time`
Definition at line 143 of file monitor_pvt.h.

12.260.3.29 valid_sats

`uint8_t Monitor_Pvt::valid_sats`
Definition at line 123 of file monitor_pvt.h.

12.260.3.30 vdop

`double Monitor_Pvt::vdop`
Definition at line 137 of file monitor_pvt.h.

12.260.3.31 vel_e

`double Monitor_Pvt::vel_e`
Definition at line 112 of file monitor_pvt.h.

12.260.3.32 vel_n

`double Monitor_Pvt::vel_n`
Definition at line 113 of file monitor_pvt.h.

12.260.3.33 vel_u

`double Monitor_Pvt::vel_u`
Definition at line 114 of file monitor_pvt.h.

12.260.3.34 vel_x

`double Monitor_Pvt::vel_x`
Definition at line 93 of file monitor_pvt.h.

12.260.3.35 vel_y

`double Monitor_Pvt::vel_y`
Definition at line 94 of file monitor_pvt.h.

12.260.3.36 vel_z

`double Monitor_Pvt::vel_z`
Definition at line 95 of file monitor_pvt.h.

12.260.3.37 week

`uint32_t Monitor_Pvt::week`
Definition at line 83 of file monitor_pvt.h.
The documentation for this class was generated from the following file:

- monitor_pvt.h

12.261 Monitor_Pvt_Udp_Sink Class Reference

Public Member Functions

- **Monitor_Pvt_Udp_Sink** (const std::vector< std::string > &addresses, const std::vector< std::string > &ports, bool protobuf_enabled)
- bool **write_monitor_pvt** (const Monitor_Pvt *const monitor_pvt)

12.261.1 Detailed Description

Definition at line 40 of file monitor_pvt_udp_sink.h.
The documentation for this class was generated from the following file:

- monitor_pvt_udp_sink.h

12.262 msm_h_t Struct Reference

Public Attributes

- unsigned char iod
- unsigned char time_s
- unsigned char clk_str
- unsigned char clk_ext
- unsigned char smooth
- unsigned char tint_s
- unsigned char nsat
- unsigned char nsig
- unsigned char sats [64]
- unsigned char sigs [32]
- unsigned char cellmask [64]

12.262.1 Detailed Description

Definition at line 1356 of file rtklib.h.

12.262.2 Member Data Documentation

12.262.2.1 cellmask

`unsigned char msm_h_t::cellmask[64]`
Definition at line 1367 of file rtklib.h.

12.262.2.2 clk_ext

`unsigned char msm_h_t::clk_ext`
Definition at line 1361 of file rtklib.h.

12.262.2.3 clk_str

`unsigned char msm_h_t::clk_str`
Definition at line 1360 of file rtklib.h.

12.262.2.4 iod

`unsigned char msm_h_t::iod`
Definition at line 1358 of file rtklib.h.

12.262.2.5 nsat

`unsigned char msm_h_t::nsat`
Definition at line 1364 of file rtklib.h.

12.262.2.6 nsig

`unsigned char msm_h_t::nsig`
Definition at line 1364 of file rtklib.h.

12.262.2.7 sats

`unsigned char msm_h_t::sats[64]`
Definition at line 1365 of file rtklib.h.

12.262.2.8 sigs

`unsigned char msm_h_t::sigs[32]`
Definition at line 1366 of file rtklib.h.

12.262.2.9 smooth

`unsigned char msm_h_t::smooth`
Definition at line 1362 of file rtklib.h.

12.262.2.10 time_s

`unsigned char msm_h_t::time_s`
Definition at line 1359 of file rtklib.h.

12.262.2.11 tint_s

`unsigned char msm_h_t::tint_s`
Definition at line 1363 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.263 mt1_header Struct Reference

Public Attributes

- uint16_t toh
- uint8_t mask_id
- uint8_t iod_set_id
- uint8_t reserved
- bool mask_flag
- bool orbit_correction_flag
- bool clock_fullset_flag
- bool clock_subset_flag
- bool code_bias_flag
- bool phase_bias_flag

12.263.1 Detailed Description

Definition at line 31 of file galileo_has_data.h.

12.263.2 Member Data Documentation

12.263.2.1 clock_fullset_flag

```
bool mt1_header::clock_fullset_flag
```

Definition at line 39 of file galileo_has_data.h.

12.263.2.2 clock_subset_flag

```
bool mt1_header::clock_subset_flag
```

Definition at line 40 of file galileo_has_data.h.

12.263.2.3 code_bias_flag

```
bool mt1_header::code_bias_flag
```

Definition at line 41 of file galileo_has_data.h.

12.263.2.4 iod_set_id

```
uint8_t mt1_header::iod_set_id
```

Definition at line 35 of file galileo_has_data.h.

12.263.2.5 mask_flag

```
bool mt1_header::mask_flag
```

Definition at line 37 of file galileo_has_data.h.

12.263.2.6 mask_id

```
uint8_t mt1_header::mask_id
```

Definition at line 34 of file galileo_has_data.h.

12.263.2.7 orbit_correction_flag

```
bool mt1_header::orbit_correction_flag
```

Definition at line 38 of file galileo_has_data.h.

12.263.2.8 phase_bias_flag

```
bool mt1_header::phase_bias_flag
```

Definition at line 42 of file galileo_has_data.h.

12.263.2.9 reserved

`uint8_t mt1_header::reserved`
Definition at line 36 of file galileo_has_data.h.

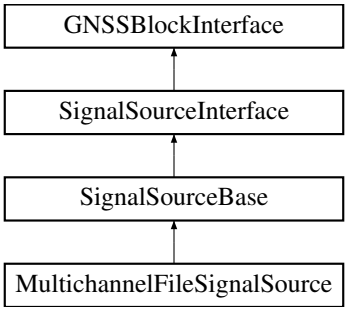
12.263.2.10 toh

`uint16_t mt1_header::toh`
Definition at line 33 of file galileo_has_data.h.
The documentation for this struct was generated from the following file:

- galileo_has_data.h

12.264 MultichannelFileSignalSource Class Reference

Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface.
`#include <multichannel_file_signal_source.h>`
Inheritance diagram for MultichannelFileSignalSource:



Public Member Functions

- **MultichannelFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const
- std::string item_type () const
- bool repeat () const
- int64_t sampling_frequency () const
- uint64_t samples () const

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.264.1 Detailed Description

Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface.
Definition at line 49 of file multichannel_file_signal_source.h.

12.264.2 Member Function Documentation

12.264.2.1 connect()

```
void MultichannelFileSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.264.2.2 disconnect()

```
void MultichannelFileSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.264.2.3 filename()

```
std::string MultichannelFileSignalSource::filename () const  [inline]
```

Definition at line 68 of file multichannel_file_signal_source.h.

12.264.2.4 get_left_block()

```
gr::basic_block_sptr MultichannelFileSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.264.2.5 get_right_block()

```
gr::basic_block_sptr MultichannelFileSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.264.2.6 item_size()

```
size_t MultichannelFileSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 58 of file multichannel_file_signal_source.h.

12.264.2.7 item_type()

```
std::string MultichannelFileSignalSource::item_type () const  [inline]
```

Definition at line 73 of file multichannel_file_signal_source.h.

12.264.2.8 repeat()

```
bool MultichannelFileSignalSource::repeat () const  [inline]
```

Definition at line 78 of file multichannel_file_signal_source.h.

12.264.2.9 samples()

```
uint64_t MultichannelFileSignalSource::samples () const  [inline]
```
Definition at line 88 of file multichannel_file_signal_source.h.

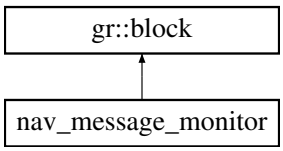
12.264.2.10 sampling_frequency()

```
int64_t MultichannelFileSignalSource::sampling_frequency () const  [inline]
```
Definition at line 83 of file multichannel_file_signal_source.h.
The documentation for this class was generated from the following file:

- multichannel_file_signal_source.h

12.265 nav_message_monitor Class Reference

GNU Radio block that receives asynchronous Nav_Message_Packet obkects from the telemetry blocks and sends them via UDP.
`#include <nav_message_monitor.h>`
Inheritance diagram for nav_message_monitor:



Public Member Functions

- ~nav_message_monitor ()=default

  Default destructor.

Friends

- nav_message_monitor_sptr **nav_message_monitor_make** (const std::vector< std::string > &addresses, uint16_t port)

12.265.1 Detailed Description

GNU Radio block that receives asynchronous Nav_Message_Packet obkects from the telemetry blocks and sends them via UDP.
Definition at line 45 of file nav_message_monitor.h.

12.265.2 Constructor & Destructor Documentation

12.265.2.1 ~nav_message_monitor()

```
nav_message_monitor::~nav_message_monitor ()  [default]
```
Default destructor.
The documentation for this class was generated from the following file:

- nav_message_monitor.h

12.266 Nav_Message_Packet Class Reference

Public Member Functions

- Nav_Message_Packet ()=default

  Default constructor.

- ~Nav_Message_Packet ()=default

  Default destructor.
- Nav_Message_Packet (const Nav_Message_Packet &other) noexcept

  Copy constructor.
- Nav_Message_Packet & operator= (const Nav_Message_Packet &rhs) noexcept

  Copy assignment operator.
- Nav_Message_Packet (Nav_Message_Packet &&other) noexcept

  Move constructor.
- Nav_Message_Packet & operator= (Nav_Message_Packet &&other) noexcept

  Move assignment operator.

Public Attributes

- std::string system

  GNSS constellation: "G" for GPS, "R" for Glonass, "S" for SBAS, "E" for Galileo and "C" for Beidou.
- std::string signal

  GNSS signal: "1C" for GPS L1 C/A, "1B" for Galileo E1b/c, "1G" for Glonass L1 C/A, "2S" for GPS L2 L2C(M), "2G" for Glonass L2 C/A, "L5" for GPS L5 and "5X" for Galileo E5a.
- int32_t prn

  SV ID.
- int32_t tow_at_current_symbol_ms

  Time of week of the current symbol, in ms.
- std::string nav_message

  Content of the navigation page.

12.266.1 Detailed Description

Definition at line 29 of file nav_message_packet.h.

12.266.2 Constructor & Destructor Documentation

12.266.2.1 Nav_Message_Packet() [1/3]

```
Nav_Message_Packet::Nav_Message_Packet () [default]
```

Default constructor.
Referenced by Nav_Message_Packet(), Nav_Message_Packet(), operator=(), and operator=().

12.266.2.2 ~Nav_Message_Packet()

```
Nav_Message_Packet::~Nav_Message_Packet () [default]
```

Default destructor.

12.266.2.3 Nav_Message_Packet() [2/3]

```
Nav_Message_Packet::Nav_Message_Packet (
            const Nav_Message_Packet & other) [inline], [noexcept]
```

Copy constructor.
Definition at line 43 of file nav_message_packet.h.
References Nav_Message_Packet().

12.266.2.4 Nav_Message_Packet() [3/3]

```
Nav_Message_Packet::Nav_Message_Packet (
            Nav_Message_Packet && other) [inline], [noexcept]
```

Move constructor.
Definition at line 64 of file nav_message_packet.h.
References Nav_Message_Packet().

12.266.3 Member Function Documentation

12.266.3.1 operator=() [1/2]

```
Nav_Message_Packet & Nav_Message_Packet::operator= (
            const Nav_Message_Packet & rhs)  [inline], [noexcept]
```

Copy assignment operator.
Definition at line 49 of file nav_message_packet.h.
References Nav_Message_Packet().

12.266.3.2 operator=() [2/2]

```
Nav_Message_Packet & Nav_Message_Packet::operator= (
            Nav_Message_Packet && other)  [inline], [noexcept]
```

Move assignment operator.
Definition at line 70 of file nav_message_packet.h.
References Nav_Message_Packet().

12.266.4 Member Data Documentation

12.266.4.1 nav_message

```
std::string Nav_Message_Packet::nav_message
```

Content of the navigation page.
Definition at line 40 of file nav_message_packet.h.
Referenced by Serdes_Nav_Message::readProtobuffer().

12.266.4.2 prn

```
int32_t Nav_Message_Packet::prn
```

SV ID.
Definition at line 38 of file nav_message_packet.h.
Referenced by Serdes_Nav_Message::readProtobuffer().

12.266.4.3 signal

```
std::string Nav_Message_Packet::signal
```

GNSS signal: "1C" for GPS L1 C/A, "1B" for Galileo E1b/c, "1G" for Glonass L1 C/A, "2S" for GPS L2 L2C(M), "2G" for Glonass L2 C/A, "L5" for GPS L5 and "5X" for Galileo E5a.
Definition at line 37 of file nav_message_packet.h.
Referenced by Serdes_Nav_Message::readProtobuffer().

12.266.4.4 system

```
std::string Nav_Message_Packet::system
```

GNSS constellation: "G" for GPS, "R" for Glonass, "S" for SBAS, "E" for Galileo and "C" for Beidou.
Definition at line 36 of file nav_message_packet.h.
Referenced by Serdes_Nav_Message::readProtobuffer().

12.266.4.5 tow_at_current_symbol_ms

```
int32_t Nav_Message_Packet::tow_at_current_symbol_ms
```

Time of week of the current symbol, in ms.
Definition at line 39 of file nav_message_packet.h.
Referenced by Serdes_Nav_Message::readProtobuffer().
The documentation for this class was generated from the following file:

- nav_message_packet.h

12.267 Nav_Message_Udp_Sink Class Reference

Public Member Functions

- **Nav_Message_Udp_Sink** (const std::vector< std::string > &addresses, const uint16_t &port)
- bool **write_nav_message** (const std::shared_ptr< Nav_Message_Packet > &nav_meg_packet)

12.267.1 Detailed Description

Definition at line 39 of file nav_message_udp_sink.h.
The documentation for this class was generated from the following file:

- nav_message_udp_sink.h

12.268 nav_t Struct Reference

Public Attributes

- int n
- int nmax
- int ng
- int ngmax
- int ns
- int nsmax
- int ne
- int nemax
- int nc
- int ncmax
- int na
- int namax
- int nt
- int ntmax
- int nf
- int nfmax
- eph_t * eph
- geph_t * geph
- seph_t * seph
- peph_t * peph
- pclk_t * pclk
- alm_t * alm
- tec_t * tec
- fcbd_t * fcb
- erp_t erp
- double utc_gps [4]
- double utc_glo [4]
- double utc_gal [4]
- double utc_qzs [4]
- double utc_cmp [4]
- double utc_irn [4]
- double utc_sbs [4]
- double ion_gps [8]
- double ion_gal [4]
- double ion_qzs [8]
- double ion_cmp [8]
- double ion_bdgim [9]
- int ion_bdgim_valid

- double ion_irn [8]
- int leaps
- double lam [MAXSAT][NFREQ]
- double cbias [MAXSAT][3]
- double rbias [MAXRCV][2][3]
- double wlbias [MAXSAT]
- double glo_cpbias [4]
- char glo_fcn [MAXPRNGLO+1]
- pcv_t pcvs [MAXSAT]
- sbssat_t sbssat
- sbsion_t sbsion [MAXBAND+1]
- dgps_t dgps [MAXSAT]
- ssr_t ssr [MAXSAT]
- lexeph_t lexeph [MAXSAT]
- lexion_t lexion
- pppcorr_t pppcorr

12.268.1 Detailed Description

Definition at line 788 of file rtklib.h.

12.268.2 Member Data Documentation

12.268.2.1 alm

```
alm_t* nav_t::alm
```

Definition at line 803 of file rtklib.h.

12.268.2.2 cbias

```
double nav_t::cbias[MAXSAT][3]
```

Definition at line 823 of file rtklib.h.

12.268.2.3 dgps

```
dgps_t nav_t::dgps[MAXSAT]
```

Definition at line 831 of file rtklib.h.

12.268.2.4 eph

```
eph_t* nav_t::eph
```

Definition at line 798 of file rtklib.h.

12.268.2.5 erp

```
erp_t nav_t::erp
```

Definition at line 806 of file rtklib.h.

12.268.2.6 fcb

```
fcbd_t* nav_t::fcb
```

Definition at line 805 of file rtklib.h.

12.268.2.7 geph

```
geph_t* nav_t::geph
```

Definition at line 799 of file rtklib.h.

12.268.2.8 glo_cpbias

```
double nav_t::glo_cpbias[4]
```
Definition at line 826 of file rtklib.h.

12.268.2.9 glo_fcn

```
char nav_t::glo_fcn[MAXPRNGLO+1]
```
Definition at line 827 of file rtklib.h.

12.268.2.10 ion_bdgim

```
double nav_t::ion_bdgim[9]
```
Definition at line 818 of file rtklib.h.

12.268.2.11 ion_bdgim_valid

```
int nav_t::ion_bdgim_valid
```
Definition at line 819 of file rtklib.h.

12.268.2.12 ion_cmp

```
double nav_t::ion_cmp[8]
```
Definition at line 817 of file rtklib.h.

12.268.2.13 ion_gal

```
double nav_t::ion_gal[4]
```
Definition at line 815 of file rtklib.h.

12.268.2.14 ion_gps

```
double nav_t::ion_gps[8]
```
Definition at line 814 of file rtklib.h.

12.268.2.15 ion_irn

```
double nav_t::ion_irn[8]
```
Definition at line 820 of file rtklib.h.

12.268.2.16 ion_qzs

```
double nav_t::ion_qzs[8]
```
Definition at line 816 of file rtklib.h.

12.268.2.17 lam

```
double nav_t::lam[MAXSAT][NFREQ]
```
Definition at line 822 of file rtklib.h.

12.268.2.18 leaps

```
int nav_t::leaps
```
Definition at line 821 of file rtklib.h.

12.268.2.19 lexeph

```
lexeph_t nav_t::lexeph[MAXSAT]
```
Definition at line 833 of file rtklib.h.

12.268.2.20 lexion

lexion_t nav_t::lexion
Definition at line 834 of file rtklib.h.

12.268.2.21 n

int nav_t::n
Definition at line 790 of file rtklib.h.

12.268.2.22 na

int nav_t::na
Definition at line 795 of file rtklib.h.

12.268.2.23 namax

int nav_t::namax
Definition at line 795 of file rtklib.h.

12.268.2.24 nc

int nav_t::nc
Definition at line 794 of file rtklib.h.

12.268.2.25 ncmax

int nav_t::ncmax
Definition at line 794 of file rtklib.h.

12.268.2.26 ne

int nav_t::ne
Definition at line 793 of file rtklib.h.

12.268.2.27 nemax

int nav_t::nemax
Definition at line 793 of file rtklib.h.

12.268.2.28 nf

int nav_t::nf
Definition at line 797 of file rtklib.h.

12.268.2.29 nfmax

int nav_t::nfmax
Definition at line 797 of file rtklib.h.

12.268.2.30 ng

int nav_t::ng
Definition at line 791 of file rtklib.h.

12.268.2.31 ngmax

int nav_t::ngmax
Definition at line 791 of file rtklib.h.

12.268.2.32 nmax

```
int nav_t::nmax
```
Definition at line 790 of file rtklib.h.

12.268.2.33 ns

```
int nav_t::ns
```
Definition at line 792 of file rtklib.h.

12.268.2.34 nsmax

```
int nav_t::nsmax
```
Definition at line 792 of file rtklib.h.

12.268.2.35 nt

```
int nav_t::nt
```
Definition at line 796 of file rtklib.h.

12.268.2.36 ntmax

```
int nav_t::ntmax
```
Definition at line 796 of file rtklib.h.

12.268.2.37 pclk

```
pclk_t* nav_t::pclk
```
Definition at line 802 of file rtklib.h.

12.268.2.38 pcvs

```
pcv_t nav_t::pcvs[MAXSAT]
```
Definition at line 828 of file rtklib.h.

12.268.2.39 peph

```
peph_t* nav_t::peph
```
Definition at line 801 of file rtklib.h.

12.268.2.40 pppcorr

```
pppcorr_t nav_t::pppcorr
```
Definition at line 835 of file rtklib.h.

12.268.2.41 rbias

```
double nav_t::rbias[MAXRCV][2][3]
```
Definition at line 824 of file rtklib.h.

12.268.2.42 sbsion

```
sbsion_t nav_t::sbsion[MAXBAND+1]
```
Definition at line 830 of file rtklib.h.

12.268.2.43 sbssat

```
sbssat_t nav_t::sbssat
```
Definition at line 829 of file rtklib.h.

12.268.2.44 seph

`seph_t* nav_t::seph`
Definition at line 800 of file rtklib.h.

12.268.2.45 ssr

`ssr_t nav_t::ssr[MAXSAT]`
Definition at line 832 of file rtklib.h.

12.268.2.46 tec

`tec_t* nav_t::tec`
Definition at line 804 of file rtklib.h.

12.268.2.47 utc_cmp

`double nav_t::utc_cmp[4]`
Definition at line 811 of file rtklib.h.

12.268.2.48 utc_gal

`double nav_t::utc_gal[4]`
Definition at line 809 of file rtklib.h.

12.268.2.49 utc_glo

`double nav_t::utc_glo[4]`
Definition at line 808 of file rtklib.h.

12.268.2.50 utc_gps

`double nav_t::utc_gps[4]`
Definition at line 807 of file rtklib.h.

12.268.2.51 utc_irn

`double nav_t::utc_irn[4]`
Definition at line 812 of file rtklib.h.

12.268.2.52 utc_qzs

`double nav_t::utc_qzs[4]`
Definition at line 810 of file rtklib.h.

12.268.2.53 utc_sbs

`double nav_t::utc_sbs[4]`
Definition at line 813 of file rtklib.h.

12.268.2.54 wlbias

`double nav_t::wlbias[MAXSAT]`
Definition at line 825 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.269 Nmea_Printer Class Reference

This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA).
`#include <nmea_printer.h>`

Public Member Functions

- Nmea_Printer (const std::string &filename, bool flag_nmea_output_file, bool flag_nmea_tty_port, std::string nmea_dump_devname, const std::string &base_path=".")

  Default constructor.
- ~Nmea_Printer () noexcept

  Default destructor.
- bool Print_Nmea_Line (const Rtklib_Solver *const pvt_data)

  Print NMEA PVT and satellite info to the initialized device.
- std::string get_GPGGA (const Rtklib_Solver *const pvt_data) const

  Returns GPGGA message (fix data).
- std::string get_GPGSA (const Rtklib_Solver *const pvt_data) const

  Returns GPGSA message (overall satellite reception data).

12.269.1 Detailed Description

This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA).
See https://en.wikipedia.org/wiki/NMEA_0183
Definition at line 44 of file nmea_printer.h.

12.269.2 Constructor & Destructor Documentation

12.269.2.1 Nmea_Printer()

```
Nmea_Printer::Nmea_Printer (
            const std::string & filename,
            bool flag_nmea_output_file,
            bool flag_nmea_tty_port,
            std::string nmea_dump_devname,
            const std::string & base_path = ".")
```

Default constructor.

12.269.2.2 ~Nmea_Printer()

```
Nmea_Printer::~Nmea_Printer ()  [noexcept]
```

Default destructor.

12.269.3 Member Function Documentation

12.269.3.1 get_GPGGA()

```
std::string Nmea_Printer::get_GPGGA (
            const Rtklib_Solver *const pvt_data) const
```

Returns GPGGA message (fix data).
References get_GPGGA().
Referenced by get_GPGGA(), and get_GPGSA().

12.269.3.2 get_GPGSA()

```
std::string Nmea_Printer::get_GPGSA (
            const Rtklib_Solver *const pvt_data) const
```

Returns GPGSA message (overall satellite reception data).
References get_GPGGA(), and get_GPGSA().
Referenced by get_GPGSA().

12.269.3.3 Print_Nmea_Line()

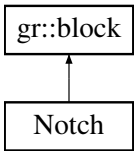
```
bool Nmea_Printer::Print_Nmea_Line (
            const Rtklib_Solver *const pvt_data)
```

Print NMEA PVT and satellite info to the initialized device.
References Print_Nmea_Line().
Referenced by Print_Nmea_Line().
The documentation for this class was generated from the following file:

- nmea_printer.h

12.270 Notch Class Reference

This class implements a real-time software-defined multi state notch filter.
`#include <notch_cc.h>`
Inheritance diagram for Notch:



Public Member Functions

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- notch_sptr **make_notch_filter** (float pfa, float p_c_factor, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)

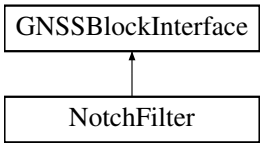
12.270.1 Detailed Description

This class implements a real-time software-defined multi state notch filter.
Definition at line 48 of file notch_cc.h.
The documentation for this class was generated from the following file:

- notch_cc.h

12.271 NotchFilter Class Reference

Inheritance diagram for NotchFilter:



Public Member Functions

- **NotchFilter** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role ()
- std::string implementation ()

  Returns "Notch_Filter".
- size_t item_size ()
- void connect (gr::top_block_sptr top_block)
- void disconnect (gr::top_block_sptr top_block)
- gr::basic_block_sptr get_left_block ()
- gr::basic_block_sptr get_right_block ()

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.271.1 Detailed Description

Definition at line 35 of file notch_filter.h.

12.271.2 Member Function Documentation

12.271.2.1 connect()

```
void NotchFilter::connect (
            gr::top_block_sptr top_block)  [virtual]
```

Implements GNSSBlockInterface.

12.271.2.2 disconnect()

```
void NotchFilter::disconnect (
            gr::top_block_sptr top_block)  [virtual]
```

Implements GNSSBlockInterface.

12.271.2.3 get_left_block()

```
gr::basic_block_sptr NotchFilter::get_left_block ()  [virtual]
```

Implements GNSSBlockInterface.

12.271.2.4 get_right_block()

```
gr::basic_block_sptr NotchFilter::get_right_block ()  [virtual]
```

Implements GNSSBlockInterface.

12.271.2.5 implementation()

```
std::string NotchFilter::implementation ()  [inline], [virtual]
```

Returns "Notch_Filter".
Implements GNSSBlockInterface.
Definition at line 50 of file notch_filter.h.

12.271.2.6 item_size()

```
size_t NotchFilter::item_size ()  [inline], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 55 of file notch_filter.h.

12.271.2.7 role()

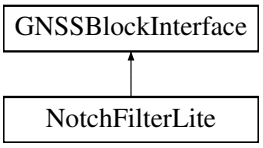
```
std::string NotchFilter::role ()  [inline], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 44 of file notch_filter.h.
The documentation for this class was generated from the following file:

- notch_filter.h

12.272 NotchFilterLite Class Reference

Inheritance diagram for NotchFilterLite:



Public Member Functions

- **NotchFilterLite** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role ()
- std::string implementation ()

  Returns "Notch_Filter_Lite".
- size_t item_size ()
- void connect (gr::top_block_sptr top_block)
- void disconnect (gr::top_block_sptr top_block)
- gr::basic_block_sptr get_left_block ()
- gr::basic_block_sptr get_right_block ()

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.272.1 Detailed Description

Definition at line 35 of file notch_filter_lite.h.

12.272.2 Member Function Documentation

12.272.2.1 connect()

```
void NotchFilterLite::connect (
            gr::top_block_sptr top_block)  [virtual]
```

Implements GNSSBlockInterface.

12.272.2.2 disconnect()

```
void NotchFilterLite::disconnect (
            gr::top_block_sptr top_block)  [virtual]
```

Implements GNSSBlockInterface.

12.272.2.3 get_left_block()

`gr::basic_block_sptr NotchFilterLite::get_left_block () [virtual]`
Implements GNSSBlockInterface.

12.272.2.4 get_right_block()

`gr::basic_block_sptr NotchFilterLite::get_right_block () [virtual]`
Implements GNSSBlockInterface.

12.272.2.5 implementation()

`std::string NotchFilterLite::implementation () [inline], [virtual]`
Returns "Notch_Filter_Lite".
Implements GNSSBlockInterface.
Definition at line 50 of file notch_filter_lite.h.

12.272.2.6 item_size()

`size_t NotchFilterLite::item_size () [inline], [virtual]`
Implements GNSSBlockInterface.
Definition at line 55 of file notch_filter_lite.h.

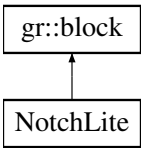
12.272.2.7 role()

`std::string NotchFilterLite::role () [inline], [virtual]`
Implements GNSSBlockInterface.
Definition at line 44 of file notch_filter_lite.h.
The documentation for this class was generated from the following file:

- notch_filter_lite.h

12.273 NotchLite Class Reference

This class implements a real-time software-defined multi state notch filter light version.
`#include <notch_lite_cc.h>`
Inheritance diagram for NotchLite:



Public Member Functions

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- notch_lite_sptr **make_notch_filter_lite** (float p_c_factor, float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset, int32_t n_segments_coeff)

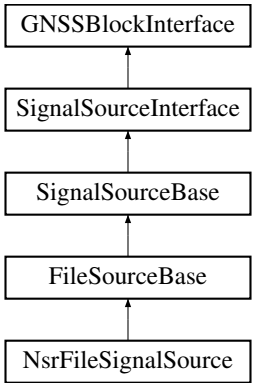
12.273.1 Detailed Description

This class implements a real-time software-defined multi state notch filter light version.
Definition at line 49 of file notch_lite_cc.h.
The documentation for this class was generated from the following file:

- notch_lite_cc.h

12.274 NsrFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <nsr_file_signal_source.h>`
Inheritance diagram for NsrFileSignalSource:



Public Member Functions

- **NsrFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in↩ _streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- double packetsPerSample () const override

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.274.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 41 of file nsr_file_signal_source.h.

12.274.2 Member Function Documentation

12.274.2.1 create_file_source_hook()

```
void NsrFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.274.2.2 itemTypeToSize()

```
std::tuple< size_t, bool > NsrFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```

Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.274.2.3 packetsPerSample()

```
double NsrFileSignalSource::packetsPerSample () const  [override], [protected], [virtual]
```

The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented from FileSourceBase.

12.274.2.4 pre_connect_hook()

```
void NsrFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.274.2.5 pre_disconnect_hook()

```
void NsrFileSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.274.2.6 source()

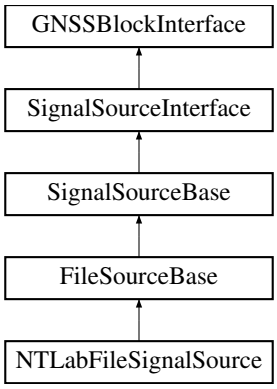
```
gnss_shared_ptr< gr::block > NsrFileSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- nsr_file_signal_source.h

12.275 NTLabFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <ntlab_file_signal_source.h>`
Inheritance diagram for NTLabFileSignalSource:



Public Member Functions

- **NTLabFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- double packetsPerSample () const override

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.275.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 43 of file ntlab_file_signal_source.h.

12.275.2 Member Function Documentation

12.275.2.1 create_file_source_hook()

```
void NTLabFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```
Reimplemented from FileSourceBase.

12.275.2.2 get_left_block()

```
gr::basic_block_sptr NTLabFileSignalSource::get_left_block ()  [override], [virtual]
```
Reimplemented from FileSourceBase.

12.275.2.3 get_right_block()

```
gr::basic_block_sptr NTLabFileSignalSource::get_right_block ()  [override], [virtual]
```
Reimplemented from FileSourceBase.

12.275.2.4 itemTypeToSize()

```
std::tuple< size_t, bool > NTLabFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```
Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.275.2.5 packetsPerSample()

```
double NTLabFileSignalSource::packetsPerSample () const  [override], [protected], [virtual]
```
The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented from FileSourceBase.

12.275.2.6 pre_connect_hook()

```
void NTLabFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```
Reimplemented from FileSourceBase.

12.275.2.7 pre_disconnect_hook()

```
void NTLabFileSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```
Reimplemented from FileSourceBase.

12.275.2.8 source()

```
gnss_shared_ptr< gr::block > NTLabFileSignalSource::source () const  [override], [protected],
[virtual]
```
Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- ntlab_file_signal_source.h

12.276 Ntrip_Rtcm_Client Class Reference

Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster, preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced v1, optionally over TLS.
`#include <ntrip_rtcm_client.h>`

Public Member Functions

- **Ntrip_Rtcm_Client** (const Ntrip_Rtcm_Client_Config &config)
- **Ntrip_Rtcm_Client** (const Ntrip_Rtcm_Client &)=delete
- Ntrip_Rtcm_Client & **operator=** (const Ntrip_Rtcm_Client &)=delete
- **Ntrip_Rtcm_Client** (Ntrip_Rtcm_Client &&)=delete
- Ntrip_Rtcm_Client & **operator=** (Ntrip_Rtcm_Client &&)=delete
- bool **start** ()
- void **stop** () noexcept
- bool **running** () const noexcept
- Ntrip_Rtcm_Client_Status **status** () const
- Ntrip_Rtcm_Snapshot **latest_snapshot** () const
- Ntrip_Rtcm_Snapshot **latest_snapshot** (const gtime_t &rover_time) const
- std::uint64_t **snapshot_generation** () const noexcept
- std::uint64_t **state_generation** () const noexcept
- void **update_rover_position** (const std::array< double, 3 > &position_ecef_m)
- void **update_rover_position** (const std::array< double, 3 > &position_ecef_m, const gtime_t &rover_time)
- void **update_rover_time** (const gtime_t &rover_time)

Friends

- class Ntrip_Rtcm_Client_Test_Access

12.276.1 Detailed Description

Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster, preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced v1, optionally over TLS.
The object owns one worker thread. The public snapshot and status methods copy data under a mutex and can be called concurrently with the worker.
Definition at line 165 of file ntrip_rtcm_client.h.

12.276.2 Friends And Related Symbol Documentation

12.276.2.1 Ntrip_Rtcm_Client_Test_Access

```
friend class Ntrip_Rtcm_Client_Test_Access  [friend]
```

Definition at line 206 of file ntrip_rtcm_client.h.
The documentation for this class was generated from the following file:

- ntrip_rtcm_client.h

12.277 Ntrip_Rtcm_Client_Config Struct Reference

Public Member Functions

- std::string validate () const

  Returns an empty string when the configuration is usable, or the reason it is not. Single source of the NTRIP configuration rules, shared by the adapter (which turns the reason into a throw at constructor time) and by start() (which turns it into an ERROR state).

Public Attributes

- bool enabled = false
- std::string host
- std::uint16_t port = 2101
- std::string mountpoint
- Secure_String username
- Secure_String password

- int reconnect_interval_ms = 10000
- int timeout_ms = 10000
- double max_age_s = 5.0
- bool send_gga = false
- int gga_period_ms = 10000
- int version = 2
- bool tls_enabled = false
- int station_id = 0

12.277.1 Detailed Description

Definition at line 48 of file ntrip_rtcm_client.h.

12.277.2 Member Function Documentation

12.277.2.1 validate()

```
std::string Ntrip_Rtcm_Client_Config::validate () const
```

Returns an empty string when the configuration is usable, or the reason it is not. Single source of the NTRIP configuration rules, shared by the adapter (which turns the reason into a throw at constructor time) and by start() (which turns it into an ERROR state).

12.277.3 Member Data Documentation

12.277.3.1 enabled

```
bool Ntrip_Rtcm_Client_Config::enabled = false
```

Definition at line 50 of file ntrip_rtcm_client.h.

12.277.3.2 gga_period_ms

```
int Ntrip_Rtcm_Client_Config::gga_period_ms = 10000
```

Definition at line 64 of file ntrip_rtcm_client.h.

12.277.3.3 host

```
std::string Ntrip_Rtcm_Client_Config::host
```

Definition at line 51 of file ntrip_rtcm_client.h.

12.277.3.4 max_age_s

```
double Ntrip_Rtcm_Client_Config::max_age_s = 5.0
```

Definition at line 62 of file ntrip_rtcm_client.h.

12.277.3.5 mountpoint

```
std::string Ntrip_Rtcm_Client_Config::mountpoint
```

Definition at line 53 of file ntrip_rtcm_client.h.

12.277.3.6 password

```
Secure_String Ntrip_Rtcm_Client_Config::password
```

Definition at line 57 of file ntrip_rtcm_client.h.

12.277.3.7 port

```
std::uint16_t Ntrip_Rtcm_Client_Config::port = 2101
```

Definition at line 52 of file ntrip_rtcm_client.h.

12.277.3.8 reconnect_interval_ms

```
int Ntrip_Rtcm_Client_Config::reconnect_interval_ms = 10000
```
Definition at line 60 of file ntrip_rtcm_client.h.

12.277.3.9 send_gga

```
bool Ntrip_Rtcm_Client_Config::send_gga = false
```
Definition at line 63 of file ntrip_rtcm_client.h.

12.277.3.10 station_id

```
int Ntrip_Rtcm_Client_Config::station_id = 0
```
Definition at line 77 of file ntrip_rtcm_client.h.

12.277.3.11 timeout_ms

```
int Ntrip_Rtcm_Client_Config::timeout_ms = 10000
```
Definition at line 61 of file ntrip_rtcm_client.h.

12.277.3.12 tls_enabled

```
bool Ntrip_Rtcm_Client_Config::tls_enabled = false
```
Definition at line 73 of file ntrip_rtcm_client.h.

12.277.3.13 username

```
Secure_String Ntrip_Rtcm_Client_Config::username
```
Definition at line 56 of file ntrip_rtcm_client.h.

12.277.3.14 version

```
int Ntrip_Rtcm_Client_Config::version = 2
```
Definition at line 69 of file ntrip_rtcm_client.h.
The documentation for this struct was generated from the following file:

- ntrip_rtcm_client.h

12.278 Ntrip_Rtcm_Client_Status Struct Reference

Public Attributes

- Ntrip_Rtcm_Client_State state = Ntrip_Rtcm_Client_State::STOPPED
- bool running = false
- bool connected = false
- std::string message
- std::uint64_t bytes_received = 0
- std::uint64_t rtcm_messages_decoded = 0
- std::uint64_t observation_epochs_decoded = 0
- std::uint64_t decoder_errors = 0
- std::uint64_t reconnect_count = 0

12.278.1 Detailed Description

Definition at line 109 of file ntrip_rtcm_client.h.

12.278.2 Member Data Documentation

12.278.2.1 bytes_received

```
std::uint64_t Ntrip_Rtcm_Client_Status::bytes_received = 0
```
Definition at line 115 of file ntrip_rtcm_client.h.

12.278.2.2 connected

```
bool Ntrip_Rtcm_Client_Status::connected = false
```
Definition at line 113 of file ntrip_rtcm_client.h.

12.278.2.3 decoder_errors

```
std::uint64_t Ntrip_Rtcm_Client_Status::decoder_errors = 0
```
Definition at line 118 of file ntrip_rtcm_client.h.

12.278.2.4 message

```
std::string Ntrip_Rtcm_Client_Status::message
```
Definition at line 114 of file ntrip_rtcm_client.h.

12.278.2.5 observation_epochs_decoded

```
std::uint64_t Ntrip_Rtcm_Client_Status::observation_epochs_decoded = 0
```
Definition at line 117 of file ntrip_rtcm_client.h.

12.278.2.6 reconnect_count

```
std::uint64_t Ntrip_Rtcm_Client_Status::reconnect_count = 0
```
Definition at line 119 of file ntrip_rtcm_client.h.

12.278.2.7 rtcm_messages_decoded

```
std::uint64_t Ntrip_Rtcm_Client_Status::rtcm_messages_decoded = 0
```
Definition at line 116 of file ntrip_rtcm_client.h.

12.278.2.8 running

```
bool Ntrip_Rtcm_Client_Status::running = false
```
Definition at line 112 of file ntrip_rtcm_client.h.

12.278.2.9 state

```
Ntrip_Rtcm_Client_State Ntrip_Rtcm_Client_Status::state = Ntrip_Rtcm_Client_State::STOPPED
```
Definition at line 111 of file ntrip_rtcm_client.h.
The documentation for this struct was generated from the following file:

- ntrip_rtcm_client.h

12.279 Ntrip_Rtcm_Snapshot Struct Reference

Public Member Functions

- void refresh_age (const gtime_t &rover_time, double max_age_s)

  Recomputes the rover-time-dependent fields age_s and fresh. A cached copy must call this per epoch: the generation covers the received data, not the passage of rover time.

Public Attributes

- bool has_observations = false
- std::vector< obsd_t > observations
- gtime_t observation_time = {0, 0.0}
- double age_s = 0.0
- bool fresh = false
- std::uint64_t generation = 0
- bool has_base_position = false
- std::array< double, 3 > base_position_ecef_m = {{0.0, 0.0, 0.0}}
- int base_position_message_type = 0
- int station_id = 0
- int itrf = 0
- double antenna_height_m = 0.0
- std::vector< eph_t > ephemerides

12.279.1 Detailed Description

Definition at line 123 of file ntrip_rtcm_client.h.

12.279.2 Member Function Documentation

12.279.2.1 refresh_age()

```
void Ntrip_Rtcm_Snapshot::refresh_age (
            const gtime_t & rover_time,
            double max_age_s)
```

Recomputes the rover-time-dependent fields age_s and fresh. A cached copy must call this per epoch: the generation covers the received data, not the passage of rover time.

12.279.3 Member Data Documentation

12.279.3.1 age_s

```
double Ntrip_Rtcm_Snapshot::age_s = 0.0
```

Definition at line 131 of file ntrip_rtcm_client.h.

12.279.3.2 antenna_height_m

```
double Ntrip_Rtcm_Snapshot::antenna_height_m = 0.0
```

Definition at line 142 of file ntrip_rtcm_client.h.

12.279.3.3 base_position_ecef_m

```
std::array<double, 3> Ntrip_Rtcm_Snapshot::base_position_ecef_m = {{0.0, 0.0, 0.0}}
```

Definition at line 138 of file ntrip_rtcm_client.h.

12.279.3.4 base_position_message_type

```
int Ntrip_Rtcm_Snapshot::base_position_message_type = 0
```

Definition at line 139 of file ntrip_rtcm_client.h.

12.279.3.5 ephemerides

```
std::vector<eph_t> Ntrip_Rtcm_Snapshot::ephemerides
```

Definition at line 148 of file ntrip_rtcm_client.h.

12.279.3.6 fresh

```
bool Ntrip_Rtcm_Snapshot::fresh = false
```
Definition at line 132 of file ntrip_rtcm_client.h.

12.279.3.7 generation

```
std::uint64_t Ntrip_Rtcm_Snapshot::generation = 0
```
Definition at line 135 of file ntrip_rtcm_client.h.

12.279.3.8 has_base_position

```
bool Ntrip_Rtcm_Snapshot::has_base_position = false
```
Definition at line 137 of file ntrip_rtcm_client.h.

12.279.3.9 has_observations

```
bool Ntrip_Rtcm_Snapshot::has_observations = false
```
Definition at line 125 of file ntrip_rtcm_client.h.

12.279.3.10 itrf

```
int Ntrip_Rtcm_Snapshot::itrf = 0
```
Definition at line 141 of file ntrip_rtcm_client.h.

12.279.3.11 observation_time

```
gtime_t Ntrip_Rtcm_Snapshot::observation_time = {0, 0.0}
```
Definition at line 127 of file ntrip_rtcm_client.h.

12.279.3.12 observations

```
std::vector<obsd_t> Ntrip_Rtcm_Snapshot::observations
```
Definition at line 126 of file ntrip_rtcm_client.h.

12.279.3.13 station_id

```
int Ntrip_Rtcm_Snapshot::station_id = 0
```
Definition at line 140 of file ntrip_rtcm_client.h.
The documentation for this struct was generated from the following file:

- ntrip_rtcm_client.h

12.280 Ntrip_Rtk_Signal Struct Reference

```
#include <ntrip_rtk_signals.h>
```

Public Attributes

- uint32_t flag

  signal_flag bit of the tracked channel signal
- int system

  RTKLIB constellation bit (SYS_XXX).
- int band_slot

  RTKLIB frequency slot the signal occupies (0-based).
- const char * signal

  two-character channel signal code (Gnss_Synchro convention)

12.280.1 Detailed Description

Single home for the signal knowledge of the NTRIP RTK path: which tracked signals it supports and, for each one, the constellation, the RTKLIB frequency slot it occupies, and the channel signal code. The adapter's channel-set validity predicate, its error message, its derived slot count and navigation-system mask, and the client's per-slot observation filter all derive from this table, and the solver cross-checks its own band mapping against it at construction — so admitting a new SIGNAL of a supported constellation is a one-row change (plus documentation and tests; a slot-sharing preference rule like BeiDou's prefer-B1C stays an explicit exception in the client). A new SYSTEM additionally needs base-side admission in the client and the solver, and navigation-system compatibility handling in the adapter.
Definition at line 46 of file ntrip_rtk_signals.h.

12.280.2 Member Data Documentation

12.280.2.1 band_slot

`int Ntrip_Rtk_Signal::band_slot`
RTKLIB frequency slot the signal occupies (0-based).
Definition at line 50 of file ntrip_rtk_signals.h.
Referenced by ntrip_rtk_is_entry_band().

12.280.2.2 flag

`uint32_t Ntrip_Rtk_Signal::flag`
signal_flag bit of the tracked channel signal
Definition at line 48 of file ntrip_rtk_signals.h.

12.280.2.3 signal

`const char* Ntrip_Rtk_Signal::signal`
two-character channel signal code (Gnss_Synchro convention)
Definition at line 51 of file ntrip_rtk_signals.h.

12.280.2.4 system

`int Ntrip_Rtk_Signal::system`
RTKLIB constellation bit (SYS_XXX).
Definition at line 49 of file ntrip_rtk_signals.h.
The documentation for this struct was generated from the following file:

- ntrip_rtk_signals.h

12.281 ntrip_t Struct Reference

Public Attributes

- int state
- int type
- int version
- int tls_enabled
- int chunked
- int chunk_state
- unsigned int chunk_remaining
- int nb
- int request_length
- int request_offset
- int request_sent
- int response_received
- int v1_retry_requested

- char url [256]
- char host [256]
- char port [16]
- char mntpnt [256]
- char user [256]
- char passwd [256]
- char str [NTRIP_MAXSTR]
- unsigned char buff [NTRIP_MAXRSP]
- unsigned char request_buff [NTRIP_MAXRSP]
- tcpcli_t * tcp
- void * tls_ctx

12.281.1 Detailed Description

Definition at line 1235 of file rtklib.h.

12.281.2 Member Data Documentation

12.281.2.1 buff

```
unsigned char ntrip_t::buff[NTRIP_MAXRSP]
```

Definition at line 1257 of file rtklib.h.

12.281.2.2 chunk_remaining

```
unsigned int ntrip_t::chunk_remaining
```

Definition at line 1243 of file rtklib.h.

12.281.2.3 chunk_state

```
int ntrip_t::chunk_state
```

Definition at line 1242 of file rtklib.h.

12.281.2.4 chunked

```
int ntrip_t::chunked
```

Definition at line 1241 of file rtklib.h.

12.281.2.5 host

```
char ntrip_t::host[256]
```

Definition at line 1251 of file rtklib.h.

12.281.2.6 mntpnt

```
char ntrip_t::mntpnt[256]
```

Definition at line 1253 of file rtklib.h.

12.281.2.7 nb

```
int ntrip_t::nb
```

Definition at line 1244 of file rtklib.h.

12.281.2.8 passwd

```
char ntrip_t::passwd[256]
```

Definition at line 1255 of file rtklib.h.

12.281.2.9 port

```
char ntrip_t::port[16]
```
Definition at line 1252 of file rtklib.h.

12.281.2.10 request_buff

```
unsigned char ntrip_t::request_buff[NTRIP_MAXRSP]
```
Definition at line 1258 of file rtklib.h.

12.281.2.11 request_length

```
int ntrip_t::request_length
```
Definition at line 1245 of file rtklib.h.

12.281.2.12 request_offset

```
int ntrip_t::request_offset
```
Definition at line 1246 of file rtklib.h.

12.281.2.13 request_sent

```
int ntrip_t::request_sent
```
Definition at line 1247 of file rtklib.h.

12.281.2.14 response_received

```
int ntrip_t::response_received
```
Definition at line 1248 of file rtklib.h.

12.281.2.15 state

```
int ntrip_t::state
```
Definition at line 1237 of file rtklib.h.

12.281.2.16 str

```
char ntrip_t::str[NTRIP_MAXSTR]
```
Definition at line 1256 of file rtklib.h.

12.281.2.17 tcp

```
tcpcli_t* ntrip_t::tcp
```
Definition at line 1259 of file rtklib.h.

12.281.2.18 tls_ctx

```
void* ntrip_t::tls_ctx
```
Definition at line 1260 of file rtklib.h.

12.281.2.19 tls_enabled

```
int ntrip_t::tls_enabled
```
Definition at line 1240 of file rtklib.h.

12.281.2.20 type

```
int ntrip_t::type
```
Definition at line 1238 of file rtklib.h.

12.281.2.21 url

```
char ntrip_t::url[256]
```
Definition at line 1250 of file rtklib.h.

12.281.2.22 user

```
char ntrip_t::user[256]
```
Definition at line 1254 of file rtklib.h.

12.281.2.23 v1_retry_requested

```
int ntrip_t::v1_retry_requested
```
Definition at line 1249 of file rtklib.h.

12.281.2.24 version

```
int ntrip_t::version
```
Definition at line 1239 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.282 Obs_Conf Class Reference

Public Attributes

- std::string dump_filename {"obs_dump.dat"}
- int32_t smoothing_factor {0}
- uint32_t nchannels_in {0U}
- uint32_t nchannels_out {0U}
- uint32_t observable_interval_ms {20U}
- bool enable_carrier_smoothing {false}
- bool always_output_gs {false}
- bool dump {false}
- bool dump_mat {false}
- bool enable_E6 {true}
- bool enable_monitor {false}

12.282.1 Detailed Description

Definition at line 30 of file obs_conf.h.

12.282.2 Member Data Documentation

12.282.2.1 always_output_gs

```
bool Obs_Conf::always_output_gs {false}
```
Definition at line 41 of file obs_conf.h.

12.282.2.2 dump

```
bool Obs_Conf::dump {false}
```
Definition at line 42 of file obs_conf.h.

12.282.2.3 dump_filename

```
std::string Obs_Conf::dump_filename {"obs_dump.dat"}
```
Definition at line 35 of file obs_conf.h.

12.282.2.4 dump_mat

```
bool Obs_Conf::dump_mat {false}
```
Definition at line 43 of file obs_conf.h.

12.282.2.5 enable_carrier_smoothing

```
bool Obs_Conf::enable_carrier_smoothing {false}
```
Definition at line 40 of file obs_conf.h.

12.282.2.6 enable_E6

```
bool Obs_Conf::enable_E6 {true}
```
Definition at line 44 of file obs_conf.h.

12.282.2.7 enable_monitor

```
bool Obs_Conf::enable_monitor {false}
```
Definition at line 45 of file obs_conf.h.

12.282.2.8 nchannels_in

```
uint32_t Obs_Conf::nchannels_in {0U}
```
Definition at line 37 of file obs_conf.h.

12.282.2.9 nchannels_out

```
uint32_t Obs_Conf::nchannels_out {0U}
```
Definition at line 38 of file obs_conf.h.

12.282.2.10 observable_interval_ms

```
uint32_t Obs_Conf::observable_interval_ms {20U}
```
Definition at line 39 of file obs_conf.h.

12.282.2.11 smoothing_factor

```
int32_t Obs_Conf::smoothing_factor {0}
```
Definition at line 36 of file obs_conf.h.
The documentation for this class was generated from the following file:

- obs_conf.h

12.283 obs_t Struct Reference

Public Attributes

- int n
- int nmax
- obsd_t * data

12.283.1 Detailed Description

Definition at line 404 of file rtklib.h.

12.283.2 Member Data Documentation

12.283.2.1 data

```
obsd_t* obs_t::data
```
Definition at line 407 of file rtklib.h.

12.283.2.2 n

`int obs_t::n`
Definition at line 406 of file rtklib.h.

12.283.2.3 nmax

`int obs_t::nmax`
Definition at line 406 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.284 obsd_t Struct Reference

Public Attributes

- gtime_t time
- unsigned char sat
- unsigned char rcv
- unsigned char SNR [NFREQ+NEXOBS]
- unsigned char LLI [NFREQ+NEXOBS]
- unsigned char code [NFREQ+NEXOBS]
- double L [NFREQ+NEXOBS]
- double P [NFREQ+NEXOBS]
- float D [NFREQ+NEXOBS]
- float Lstd [NFREQ+NEXOBS]
- float Pstd [NFREQ+NEXOBS]

12.284.1 Detailed Description

Definition at line 389 of file rtklib.h.

12.284.2 Member Data Documentation

12.284.2.1 code

`unsigned char obsd_t::code[NFREQ+NEXOBS]`
Definition at line 395 of file rtklib.h.

12.284.2.2 D

`float obsd_t::D[NFREQ+NEXOBS]`
Definition at line 398 of file rtklib.h.

12.284.2.3 L

`double obsd_t::L[NFREQ+NEXOBS]`
Definition at line 396 of file rtklib.h.

12.284.2.4 LLI

`unsigned char obsd_t::LLI[NFREQ+NEXOBS]`
Definition at line 394 of file rtklib.h.

12.284.2.5 Lstd

`float obsd_t::Lstd[NFREQ+NEXOBS]`
Definition at line 399 of file rtklib.h.

12.284.2.6 P

`double obsd_t::P[NFREQ+NEXOBS]`
Definition at line 397 of file rtklib.h.

12.284.2.7 Pstd

`float obsd_t::Pstd[NFREQ+NEXOBS]`
Definition at line 400 of file rtklib.h.

12.284.2.8 rcv

`unsigned char obsd_t::rcv`
Definition at line 392 of file rtklib.h.

12.284.2.9 sat

`unsigned char obsd_t::sat`
Definition at line 392 of file rtklib.h.

12.284.2.10 SNR

`unsigned char obsd_t::SNR[NFREQ+NEXOBS]`
Definition at line 393 of file rtklib.h.

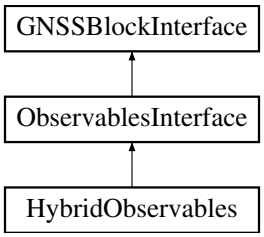
12.284.2.11 time

`gtime_t obsd_t::time`
Definition at line 391 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.285 ObservablesInterface Class Reference

This abstract class represents an interface to an observables block.
`#include <observables_interface.h>`
Inheritance diagram for ObservablesInterface:



Public Member Functions

- virtual void **reset** ()=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0

- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.285.1 Detailed Description

This abstract class represents an interface to an observables block.
Abstract class for pseudorange_intefaces, derived from GNSSBlockInterface. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 43 of file observables_interface.h.
The documentation for this class was generated from the following file:

- observables_interface.h

12.286 opt_t Struct Reference

Public Attributes

- const char * name
- int format
- void * var
- const char * comment

12.286.1 Detailed Description

Definition at line 956 of file rtklib.h.

12.286.2 Member Data Documentation

12.286.2.1 comment

```
const char* opt_t::comment
```

Definition at line 961 of file rtklib.h.

12.286.2.2 format

```
int opt_t::format
```

Definition at line 959 of file rtklib.h.

12.286.2.3 name

```
const char* opt_t::name
```

Definition at line 958 of file rtklib.h.

12.286.2.4 var

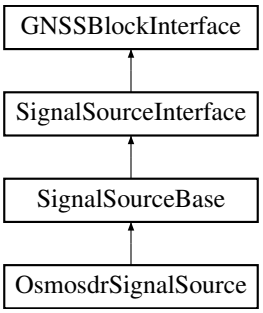
```
void* opt_t::var
```

Definition at line 960 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.287 OsmosdrSignalSource Class Reference

This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki).
`#include <osmosdr_signal_source.h>`
Inheritance diagram for OsmosdrSignalSource:



Public Member Functions

- **OsmosdrSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.287.1 Detailed Description

This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki).
Definition at line 45 of file osmosdr_signal_source.h.

12.287.2 Member Function Documentation

12.287.2.1 connect()

```
void OsmosdrSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.287.2.2 disconnect()

```
void OsmosdrSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.287.2.3 get_left_block()

```
gr::basic_block_sptr OsmosdrSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.287.2.4 get_right_block()

```
gr::basic_block_sptr OsmosdrSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.287.2.5 item_size()

```
size_t OsmosdrSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 54 of file osmosdr_signal_source.h.
The documentation for this class was generated from the following file:

- osmosdr_signal_source.h

12.288 Osnma_Crypto_Material_Manager Class Reference

Public Member Functions

- **Osnma_Crypto_Material_Manager** (const std::string &cache_dir)
- Osnma_Public_Key_Material **load_active_public_key_cache** (const std::string &expected_fingerprint=std::↩ string()) const
- Osnma_Public_Key_Material **load_public_key_metadata** (const std::string &pem_path, const std::string &expected_fingerprint=std::string()) const
- Osnma_Public_Key_Material **candidate_public_key** (uint8_t pkid) const
- Osnma_Merkle_Tree_Material **candidate_merkle_tree** () const
- Osnma_Merkle_Tree_Material **load_configured_merkle_tree** (const std::string &path) const
- Osnma_Merkle_Tree_Material **load_active_merkle_tree_cache** () const
- Osnma_Merkle_Tree_Material **load_future_merkle_tree_cache** () const
- Osnma_Public_Key_Material **active_public_key** () const
- Osnma_Merkle_Tree_Material **active_merkle_tree** () const
- bool **store_active_public_key_cache** (const Osnma_Public_Key_Material &key) const
- bool **store_public_key_metadata** (const Osnma_Public_Key_Material &key) const
- bool **store_active_merkle_tree_cache** (const Osnma_Merkle_Tree_Material &tree) const
- bool **store_future_merkle_tree_cache** (const Osnma_Merkle_Tree_Material &tree) const
- bool **has_candidate_material_for_pkid** (uint8_t pkid) const
- void **set_active_public_key** (const Osnma_Public_Key_Material &key)
- void **set_active_merkle_tree** (const Osnma_Merkle_Tree_Material &tree)
- void **set_candidate_public_key** (const Osnma_Public_Key_Material &key)
- void **set_candidate_merkle_tree** (const Osnma_Merkle_Tree_Material &tree)

- void **clear_candidate_public_key** ()
- void **clear_candidate_merkle_tree** ()
- bool **promote_candidate_for_pkid** (uint8_t pkid)
- bool **promote_candidate_merkle_tree** ()

12.288.1 Detailed Description

Definition at line 69 of file osnma_crypto_material.h.
The documentation for this class was generated from the following file:

- osnma_crypto_material.h

12.289 OSNMA_data Class Reference

This class handles ONSMA data See https://www.gsc-europa.eu/sites/default/files/↩sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf.
`#include <osnma_data.h>`

Public Attributes

- DSM_nma_header d_nma_header
- DSM_dsm_header d_dsm_header
- DSM_PKR_message d_dsm_pkr_message
- DSM_KROOT_message d_dsm_kroot_message
- DSM_KROOT_message d_dsm_kroot_new_message
- MACK_message d_mack_message
- OSNMA_NavData d_nav_data

12.289.1 Detailed Description

This class handles ONSMA data See https://www.gsc-europa.eu/sites/default/files/↩sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf.
Definition at line 237 of file osnma_data.h.

12.289.2 Member Data Documentation

12.289.2.1 d_dsm_header

`DSM_dsm_header OSNMA_data::d_dsm_header`
Definition at line 242 of file osnma_data.h.

12.289.2.2 d_dsm_kroot_message

`DSM_KROOT_message OSNMA_data::d_dsm_kroot_message`
Definition at line 244 of file osnma_data.h.

12.289.2.3 d_dsm_kroot_new_message

`DSM_KROOT_message OSNMA_data::d_dsm_kroot_new_message`
Definition at line 245 of file osnma_data.h.

12.289.2.4 d_dsm_pkr_message

`DSM_PKR_message OSNMA_data::d_dsm_pkr_message`
Definition at line 243 of file osnma_data.h.

12.289.2.5 d_mack_message

`MACK_message OSNMA_data::d_mack_message`
Definition at line 246 of file osnma_data.h.

12.289.2.6 d_nav_data

`OSNMA_NavData OSNMA_data::d_nav_data`
Definition at line 247 of file osnma_data.h.

12.289.2.7 d_nma_header

`DSM_nma_header OSNMA_data::d_nma_header`
Definition at line 241 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.290 OSNMA_DSM_Reader Class Reference

Public Member Functions

- uint8_t **get_nmas** (uint8_t nma_header) const
- uint8_t **get_cid** (uint8_t nma_header) const
- uint8_t **get_cpks** (uint8_t nma_header) const
- bool **get_nma_header_reserved** (uint8_t nma_header) const
- uint8_t **get_dsm_id** (uint8_t dsm_header) const
- uint8_t **get_dsm_block_id** (uint8_t dsm_header) const
- uint8_t **get_number_blocks_index** (uint8_t dsm_msg_0) const
- uint8_t **get_pkid** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_cidkr** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_dsm_reserved1** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_hf** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_mf** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_ks** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_ts** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_maclt** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_dsm_reserved** (const std::vector< uint8_t > &dsm_msg) const
- uint16_t **get_wn_k** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_towh_k** (const std::vector< uint8_t > &dsm_msg) const
- uint64_t **get_alpha** (const std::vector< uint8_t > &dsm_msg) const
- uint16_t **get_l_dk_bits** (uint8_t nb_dk) const
- uint16_t **get_lk_bits** (uint8_t ks) const
- std::vector< uint8_t > **get_kroot** (const std::vector< uint8_t > &dsm_msg, uint16_t bytes_lk) const
- std::string **get_hash_function** (uint8_t hf) const
- std::string **get_nmas_status** (uint8_t nmas) const
- std::string **get_cpks_status** (uint8_t cpks) const
- uint8_t **get_mid** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_npkt** (const std::vector< uint8_t > &dsm_msg) const
- uint8_t **get_npktid** (const std::vector< uint8_t > &dsm_msg) const

12.290.1 Detailed Description

Definition at line 29 of file osnma_dsm_reader.h.
The documentation for this class was generated from the following file:

- osnma_dsm_reader.h

12.291 Osnma_Helper Class Reference

Public Member Functions

- uint32_t **compute_gst** (uint32_t WN, uint32_t TOW) const
- uint32_t **compute_gst** (std::tm &input)
- uint32_t **compute_gst_now** () const
- uint32_t **get_WN** (uint32_t GST) const
- uint32_t **get_TOW** (uint32_t GST) const
- std::vector< uint8_t > **gst_to_uint8** (uint32_t GST) const
- std::vector< uint8_t > **bytes** (const std::string &binaryString) const
- std::string **verification_status_str** (int status) const
- std::string **convert_to_hex_string** (const std::vector< uint8_t > &vector) const
- std::vector< uint8_t > **convert_from_hex_string** (const std::string &hex_string) const

Public Attributes

- std::tm GST_START_EPOCH {}

12.291.1 Detailed Description

Definition at line 31 of file osnma_helper.h.

12.291.2 Member Data Documentation

12.291.2.1 GST_START_EPOCH

```
std::tm Osnma_Helper::GST_START_EPOCH {}
```

Definition at line 46 of file osnma_helper.h.
The documentation for this class was generated from the following file:

- osnma_helper.h

12.292 Osnma_Merkle_Tree_Material Struct Reference

Classes

- struct PublicKeyEntry

Public Attributes

- bool valid {false}
- std::string source
- std::string uid
- std::string applicability
- std::string state
- std::string hash_function
- std::vector< uint8_t > root
- std::string xml_path
- std::vector< PublicKeyEntry > public_keys

12.292.1 Detailed Description

Definition at line 42 of file osnma_crypto_material.h.

12.292.2 Member Data Documentation

12.292.2.1 applicability

`std::string Osnma_Merkle_Tree_Material::applicability`
Definition at line 57 of file osnma_crypto_material.h.

12.292.2.2 hash_function

`std::string Osnma_Merkle_Tree_Material::hash_function`
Definition at line 59 of file osnma_crypto_material.h.

12.292.2.3 public_keys

`std::vector<PublicKeyEntry> Osnma_Merkle_Tree_Material::public_keys`
Definition at line 62 of file osnma_crypto_material.h.

12.292.2.4 root

`std::vector<uint8_t> Osnma_Merkle_Tree_Material::root`
Definition at line 60 of file osnma_crypto_material.h.

12.292.2.5 source

`std::string Osnma_Merkle_Tree_Material::source`
Definition at line 55 of file osnma_crypto_material.h.

12.292.2.6 state

`std::string Osnma_Merkle_Tree_Material::state`
Definition at line 58 of file osnma_crypto_material.h.

12.292.2.7 uid

`std::string Osnma_Merkle_Tree_Material::uid`
Definition at line 56 of file osnma_crypto_material.h.

12.292.2.8 valid

`bool Osnma_Merkle_Tree_Material::valid {false}`
Definition at line 54 of file osnma_crypto_material.h.

12.292.2.9 xml_path

`std::string Osnma_Merkle_Tree_Material::xml_path`
Definition at line 61 of file osnma_crypto_material.h.
The documentation for this struct was generated from the following file:

- osnma_crypto_material.h

12.293 OSNMA_msg Class Reference

This class fills the OSNMA_msg structure with the data received from the telemetry blocks.
`#include <galileo_inav_message.h>`

Public Attributes

- std::array< uint32_t, 15 > mack {}
- std::array< uint8_t, 15 > hkroot {}
- std::array< uint8_t, 15 > page_valid {}

- bool page_validity_available {false}
- uint32_t PRN {}
- uint32_t WN_sf0 {}
- uint32_t TOW_sf0 {}

12.293.1 Detailed Description

This class fills the OSNMA_msg structure with the data received from the telemetry blocks.
Definition at line 47 of file galileo_inav_message.h.

12.293.2 Member Data Documentation

12.293.2.1 hkroot

```
std::array<uint8_t, 15> OSNMA_msg::hkroot {}
```
Definition at line 52 of file galileo_inav_message.h.

12.293.2.2 mack

```
std::array<uint32_t, 15> OSNMA_msg::mack {}
```
Definition at line 51 of file galileo_inav_message.h.

12.293.2.3 page_valid

```
std::array<uint8_t, 15> OSNMA_msg::page_valid {}
```
Definition at line 53 of file galileo_inav_message.h.

12.293.2.4 page_validity_available

```
bool OSNMA_msg::page_validity_available {false}
```
Definition at line 54 of file galileo_inav_message.h.

12.293.2.5 PRN

```
uint32_t OSNMA_msg::PRN {}
```
Definition at line 55 of file galileo_inav_message.h.

12.293.2.6 TOW_sf0

```
uint32_t OSNMA_msg::TOW_sf0 {}
```
Definition at line 57 of file galileo_inav_message.h.

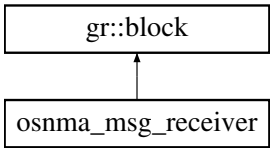
12.293.2.7 WN_sf0

```
uint32_t OSNMA_msg::WN_sf0 {}
```
Definition at line 56 of file galileo_inav_message.h.
The documentation for this class was generated from the following file:

- galileo_inav_message.h

12.294 osnma_msg_receiver Class Reference

GNU Radio block that receives asynchronous OSNMA messages from the telemetry blocks, stores them in memory, and decodes OSNMA info when enough data have been received. The decoded OSNMA data is sent to the PVT block.
`#include <osnma_msg_receiver.h>`
Inheritance diagram for osnma_msg_receiver:



Public Member Functions

- ~osnma_msg_receiver ()=default

  Default destructor.
- bool verify_dsm_pkr (const DSM_PKR_message &message) const

  Public for benchmarking purposes.
- bool **verify_dsm_pkr** (const DSM_PKR_message &message, const Osnma_Merkle_Tree_Material &merkle_tree) const
- void msg_handler_osnma (const pmt::pmt_t &msg)

  For testing purposes.
- void read_merkle_xml (const std::string &merklepath)

  Public for testing purposes.
- void set_merkle_root (const std::vector< uint8_t > &v)

  Public for benchmarking purposes.

Friends

- class OsnmaTestVectors
- osnma_msg_receiver_sptr **osnma_msg_receiver_make** (const std::string &pemFilePath, const std::string &merkleFilePath, bool strict_mode=false, bool replay_mode=false)

12.294.1 Detailed Description

GNU Radio block that receives asynchronous OSNMA messages from the telemetry blocks, stores them in memory, and decodes OSNMA info when enough data have been received. The decoded OSNMA data is sent to the PVT block.
Definition at line 64 of file osnma_msg_receiver.h.

12.294.2 Constructor & Destructor Documentation

12.294.2.1 ~osnma_msg_receiver()

```
osnma_msg_receiver::~osnma_msg_receiver ()  [default]
```

Default destructor.

12.294.3 Member Function Documentation

12.294.3.1 msg_handler_osnma()

```
void osnma_msg_receiver::msg_handler_osnma (
            const pmt::pmt_t & msg)
```

For testing purposes.

12.294.3.2 read_merkle_xml()

```
void osnma_msg_receiver::read_merkle_xml (
            const std::string & merklepath)
```

Public for testing purposes.

12.294.3.3 set_merkle_root()

```
void osnma_msg_receiver::set_merkle_root (
            const std::vector< uint8_t > & v)
```

Public for benchmarking purposes.

12.294.3.4 verify_dsm_pkr()

```
bool osnma_msg_receiver::verify_dsm_pkr (
            const DSM_PKR_message & message) const
```

Public for benchmarking purposes.
References verify_dsm_pkr().
Referenced by verify_dsm_pkr().

12.294.4 Friends And Related Symbol Documentation

12.294.4.1 OsnmaTestVectors

```
friend class OsnmaTestVectors  [friend]
```

Definition at line 374 of file osnma_msg_receiver.h.
The documentation for this class was generated from the following file:

- osnma_msg_receiver.h

12.295 OSNMA_NavData Class Reference

Public Member Functions

- std::string **get_utc_data** () const
- std::string **get_ephemeris_data** () const
- uint32_t get_verified_bits () const
- uint32_t get_prn_d () const
- uint32_t get_IOD_nav () const
- uint32_t get_wn_sf0 () const
- uint32_t get_last_received_WN () const
- uint32_t get_last_received_TOW () const
- uint32_t get_tow_sf0 () const
- uint64_t get_first_accumulated_tag_gst () const
- uint64_t get_last_accumulated_tag_gst () const
- uint8_t get_accumulated_tag_adkd () const
- bool **have_this_bits** (std::string nav_data)
- bool get_verified_status () const
- bool has_tag_accumulation () const
- bool **add_nav_data** (const std::string &nav_data)
- void set_tow_sf0 (int value)
- void set_wn_sf0 (uint32_t WN)
- void set_ephemeris_data (std::string value)
- void set_utc_data (std::string value)
- void **update_last_received_timestamp** (uint32_t TOW)
- void set_prn_d (uint32_t value)
- void set_last_received_WN (uint32_t WN)
- void set_last_received_TOW (uint32_t TOW)
- void set_update_verified_bits (uint32_t morebits)
- void set_verified_status (bool value)
- void set_IOD_nav (uint32_t value)
- void start_tag_accumulation (uint32_t bits, uint8_t adkd, uint64_t tag_gst)
- void continue_tag_accumulation (uint32_t morebits, uint64_t tag_gst)
- void clear_tag_accumulation_metadata ()
- void reset_tag_accumulation ()

Public Attributes

- const uint32_t nav_data_id

12.295.1 Detailed Description

Definition at line 156 of file osnma_data.h.

12.295.2 Constructor & Destructor Documentation

12.295.2.1 OSNMA_NavData()

```
OSNMA_NavData::OSNMA_NavData ()  [inline]
```

Definition at line 159 of file osnma_data.h.

12.295.3 Member Function Documentation

12.295.3.1 clear_tag_accumulation_metadata()

```
void OSNMA_NavData::clear_tag_accumulation_metadata ()  [inline]
```

Definition at line 201 of file osnma_data.h.

12.295.3.2 continue_tag_accumulation()

```
void OSNMA_NavData::continue_tag_accumulation (
            uint32_t morebits,
            uint64_t tag_gst)  [inline]
```

Definition at line 196 of file osnma_data.h.

12.295.3.3 get_accumulated_tag_adkd()

```
uint8_t OSNMA_NavData::get_accumulated_tag_adkd () const  [inline]
```

Definition at line 172 of file osnma_data.h.

12.295.3.4 get_first_accumulated_tag_gst()

```
uint64_t OSNMA_NavData::get_first_accumulated_tag_gst () const  [inline]
```

Definition at line 170 of file osnma_data.h.

12.295.3.5 get_IOD_nav()

```
uint32_t OSNMA_NavData::get_IOD_nav () const  [inline]
```

Definition at line 165 of file osnma_data.h.

12.295.3.6 get_last_accumulated_tag_gst()

```
uint64_t OSNMA_NavData::get_last_accumulated_tag_gst () const  [inline]
```

Definition at line 171 of file osnma_data.h.

12.295.3.7 get_last_received_TOW()

```
uint32_t OSNMA_NavData::get_last_received_TOW () const  [inline]
```

Definition at line 168 of file osnma_data.h.

12.295.3.8 get_last_received_WN()

```
uint32_t OSNMA_NavData::get_last_received_WN () const  [inline]
```

Definition at line 167 of file osnma_data.h.

12.295.3.9 get_prn_d()

```
uint32_t OSNMA_NavData::get_prn_d () const  [inline]
```
Definition at line 164 of file osnma_data.h.

12.295.3.10 get_tow_sf0()

```
uint32_t OSNMA_NavData::get_tow_sf0 () const  [inline]
```
Definition at line 169 of file osnma_data.h.

12.295.3.11 get_verified_bits()

```
uint32_t OSNMA_NavData::get_verified_bits () const  [inline]
```
Definition at line 163 of file osnma_data.h.

12.295.3.12 get_verified_status()

```
bool OSNMA_NavData::get_verified_status () const  [inline]
```
Definition at line 174 of file osnma_data.h.

12.295.3.13 get_wn_sf0()

```
uint32_t OSNMA_NavData::get_wn_sf0 () const  [inline]
```
Definition at line 166 of file osnma_data.h.

12.295.3.14 has_tag_accumulation()

```
bool OSNMA_NavData::has_tag_accumulation () const  [inline]
```
Definition at line 175 of file osnma_data.h.

12.295.3.15 reset_tag_accumulation()

```
void OSNMA_NavData::reset_tag_accumulation ()  [inline]
```
Definition at line 208 of file osnma_data.h.

12.295.3.16 set_ephemeris_data()

```
void OSNMA_NavData::set_ephemeris_data (
            std::string value)  [inline]
```
Definition at line 179 of file osnma_data.h.

12.295.3.17 set_IOD_nav()

```
void OSNMA_NavData::set_IOD_nav (
            uint32_t value)  [inline]
```
Definition at line 187 of file osnma_data.h.

12.295.3.18 set_last_received_TOW()

```
void OSNMA_NavData::set_last_received_TOW (
            uint32_t TOW)  [inline]
```
Definition at line 184 of file osnma_data.h.

12.295.3.19 set_last_received_WN()

```
void OSNMA_NavData::set_last_received_WN (
            uint32_t WN)  [inline]
```
Definition at line 183 of file osnma_data.h.

12.295.3.20 set_prn_d()

```
void OSNMA_NavData::set_prn_d (
            uint32_t value)  [inline]
```

Definition at line 182 of file osnma_data.h.

12.295.3.21 set_tow_sf0()

```
void OSNMA_NavData::set_tow_sf0 (
            int value)  [inline]
```

Definition at line 177 of file osnma_data.h.

12.295.3.22 set_update_verified_bits()

```
void OSNMA_NavData::set_update_verified_bits (
            uint32_t morebits)  [inline]
```

Definition at line 185 of file osnma_data.h.

12.295.3.23 set_utc_data()

```
void OSNMA_NavData::set_utc_data (
            std::string value)  [inline]
```

Definition at line 180 of file osnma_data.h.

12.295.3.24 set_verified_status()

```
void OSNMA_NavData::set_verified_status (
            bool value)  [inline]
```

Definition at line 186 of file osnma_data.h.

12.295.3.25 set_wn_sf0()

```
void OSNMA_NavData::set_wn_sf0 (
            uint32_t WN)  [inline]
```

Definition at line 178 of file osnma_data.h.

12.295.3.26 start_tag_accumulation()

```
void OSNMA_NavData::start_tag_accumulation (
            uint32_t bits,
            uint8_t adkd,
            uint64_t tag_gst)  [inline]
```

Definition at line 188 of file osnma_data.h.

12.295.4 Member Data Documentation

12.295.4.1 nav_data_id

```
const uint32_t OSNMA_NavData::nav_data_id
```

Definition at line 160 of file osnma_data.h.

The documentation for this class was generated from the following file:

- osnma_data.h

12.296 OSNMA_NavDataManager Class Reference

Class for managing OSNMA navigation data.

`#include <osnma_nav_data_manager.h>`

Public Member Functions

- void **log_status** () const
- bool **have_nav_data** (const Tag &t) const
- std::string **get_navigation_data** (const Tag &t) const
- void **add_navigation_data** (const std::string &nav_bits, uint32_t PRNd, uint32_t WN, uint32_t TOW)
- void **update_nav_data** (const std::multimap< uint32_t, Tag > &tags_verified, uint8_t tag_size)
- void **reset_tag_accumulations** ()
- void **prune_old_navigation_data** (uint32_t WN, uint32_t TOW)
- bool **have_nav_data** (const std::string &nav_bits, uint32_t PRNd, uint32_t WN, uint32_t TOW)
- std::vector< OSNMA_NavData > **get_verified_data** ()

12.296.1 Detailed Description

Class for managing OSNMA navigation data.
Definition at line 35 of file osnma_nav_data_manager.h.
The documentation for this class was generated from the following file:

- osnma_nav_data_manager.h

12.297 Osnma_Public_Key_Material Struct Reference

Public Attributes

- bool valid {false}
- bool pkid_valid {false}
- bool npkt_valid {false}
- uint8_t pkid {0}
- uint8_t npkt {0}
- std::string key_type
- std::string source
- std::string fingerprint_sha256
- std::string pem_path
- std::string product_uid
- std::string applicability
- std::string state
- std::vector< uint8_t > compressed_key

12.297.1 Detailed Description

Definition at line 24 of file osnma_crypto_material.h.

12.297.2 Member Data Documentation

12.297.2.1 applicability

```
std::string Osnma_Public_Key_Material::applicability
```

Definition at line 36 of file osnma_crypto_material.h.

12.297.2.2 compressed_key

```
std::vector<uint8_t> Osnma_Public_Key_Material::compressed_key
```

Definition at line 38 of file osnma_crypto_material.h.

12.297.2.3 fingerprint_sha256

```
std::string Osnma_Public_Key_Material::fingerprint_sha256
```

Definition at line 33 of file osnma_crypto_material.h.

12.297.2.4 key_type

`std::string Osnma_Public_Key_Material::key_type`
Definition at line 31 of file osnma_crypto_material.h.

12.297.2.5 npkt

`uint8_t Osnma_Public_Key_Material::npkt {0}`
Definition at line 30 of file osnma_crypto_material.h.

12.297.2.6 npkt_valid

`bool Osnma_Public_Key_Material::npkt_valid {false}`
Definition at line 28 of file osnma_crypto_material.h.

12.297.2.7 pem_path

`std::string Osnma_Public_Key_Material::pem_path`
Definition at line 34 of file osnma_crypto_material.h.

12.297.2.8 pkid

`uint8_t Osnma_Public_Key_Material::pkid {0}`
Definition at line 29 of file osnma_crypto_material.h.

12.297.2.9 pkid_valid

`bool Osnma_Public_Key_Material::pkid_valid {false}`
Definition at line 27 of file osnma_crypto_material.h.

12.297.2.10 product_uid

`std::string Osnma_Public_Key_Material::product_uid`
Definition at line 35 of file osnma_crypto_material.h.

12.297.2.11 source

`std::string Osnma_Public_Key_Material::source`
Definition at line 32 of file osnma_crypto_material.h.

12.297.2.12 state

`std::string Osnma_Public_Key_Material::state`
Definition at line 37 of file osnma_crypto_material.h.

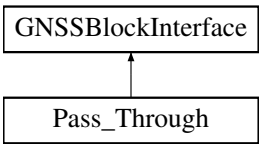
12.297.2.13 valid

`bool Osnma_Public_Key_Material::valid {false}`
Definition at line 26 of file osnma_crypto_material.h.
The documentation for this struct was generated from the following file:

- osnma_crypto_material.h

12.298 Pass_Through Class Reference

This class implements a block that connects input and output (does nothing).
`#include <pass_through.h>`
Inheritance diagram for Pass_Through:



Public Member Functions

- **Pass_Through** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream)
- std::string role () override
- std::string implementation () override

  Returns "Pass_Through".
- std::string item_type () const
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.298.1 Detailed Description

This class implements a block that connects input and output (does nothing).
Definition at line 42 of file pass_through.h.

12.298.2 Member Function Documentation

12.298.2.1 connect()

```
void Pass_Through::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.298.2.2 disconnect()

```
void Pass_Through::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.298.2.3 get_left_block()

```
gr::basic_block_sptr Pass_Through::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.298.2.4 get_right_block()

```
gr::basic_block_sptr Pass_Through::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.298.2.5 implementation()

```
std::string Pass_Through::implementation ()  [inline], [override], [virtual]
```
Returns "Pass_Through".
Implements GNSSBlockInterface.
Definition at line 58 of file pass_through.h.

12.298.2.6 item_size()

```
size_t Pass_Through::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 68 of file pass_through.h.

12.298.2.7 item_type()

```
std::string Pass_Through::item_type () const  [inline]
```
Definition at line 63 of file pass_through.h.

12.298.2.8 role()

```
std::string Pass_Through::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 52 of file pass_through.h.
The documentation for this class was generated from the following file:

- pass_through.h

12.299 pclk_t Struct Reference

Public Attributes

- gtime_t time
- int index
- double clk [MAXSAT][1]
- float std [MAXSAT][1]

12.299.1 Detailed Description

Definition at line 527 of file rtklib.h.

12.299.2 Member Data Documentation

12.299.2.1 clk

```
double pclk_t::clk[MAXSAT][1]
```
Definition at line 531 of file rtklib.h.

12.299.2.2 index

```
int pclk_t::index
```
Definition at line 530 of file rtklib.h.

12.299.2.3 std

```
float pclk_t::std[MAXSAT][1]
```
Definition at line 532 of file rtklib.h.

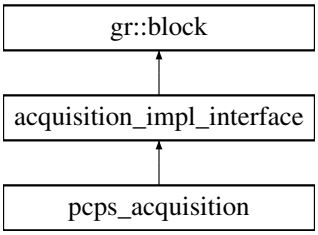
12.299.2.4 time

gtime_t pclk_t::time
Definition at line 529 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.300 pcps_acquisition Class Reference

This class implements a Parallel Code Phase Search Acquisition.
#include <pcps_acquisition.h>
Inheritance diagram for pcps_acquisition:



Public Member Functions

- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- void set_local_code (std::complex< float > *code) override

  Sets local code for PCPS acquisition algorithm.
- void **set_resampler_latency** (uint32_t latency_samples)
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- void set_doppler_center (int32_t doppler_center)

  Set Doppler center frequency for the grid search. It will refresh the Doppler grid.
- void set_doppler_num_bins (uint32_t num_doppler_bins)

  Sets how many Doppler bins to search, centered on set_doppler_center(). Refreshes the Doppler grid and updates the detection threshold when the bin count changes, supporting both unassisted and assisted searches.
- bool cuda_ready () const

  Whether the CUDA engine is available. Inspect only while acquisition is stopped.
- uint64_t cuda_grid_count () const

  Completed CUDA grids over the block's lifetime, excluding warm-up. Inspect only while acquisition is stopped.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_↩ signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_acquisition_sptr **pcps_make_acquisition** (const Acq_Conf &conf_)

12.300.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Check An Open Source Galileo E1 Software Receiver, Algorithm 1, for a pseudocode description of this implementation.
Definition at line 96 of file pcps_acquisition.h.

12.300.2 Member Function Documentation

12.300.2.1 cuda_grid_count()

```
uint64_t pcps_acquisition::cuda_grid_count () const  [inline]
```

Completed CUDA grids over the block's lifetime, excluding warm-up. Inspect only while acquisition is stopped.
Definition at line 180 of file pcps_acquisition.h.

12.300.2.2 cuda_ready()

```
bool pcps_acquisition::cuda_ready () const  [inline]
```

Whether the CUDA engine is available. Inspect only while acquisition is stopped.
Definition at line 169 of file pcps_acquisition.h.

12.300.2.3 general_work()

```
int pcps_acquisition::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.300.2.4 mag()

```
uint32_t pcps_acquisition::mag () const  [inline], [override], [virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 123 of file pcps_acquisition.h.

12.300.2.5 set_active()

```
void pcps_acquisition::set_active (
            bool active)  [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.

12.300.2.6 set_channel()

```
void pcps_acquisition::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 139 of file pcps_acquisition.h.

12.300.2.7 set_channel_fsm()

```
void pcps_acquisition::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 147 of file pcps_acquisition.h.

12.300.2.8 set_doppler_center()

```
void pcps_acquisition::set_doppler_center (
            int32_t doppler_center)
```

Set Doppler center frequency for the grid search. It will refresh the Doppler grid.

Parameters

doppler_center	- Frequency center of the search grid [Hz].

12.300.2.9 set_doppler_num_bins()

```
void pcps_acquisition::set_doppler_num_bins (
            uint32_t num_doppler_bins)
```

Sets how many Doppler bins to search, centered on set_doppler_center(). Refreshes the Doppler grid and updates the detection threshold when the bin count changes, supporting both unassisted and assisted searches.

Parameters

num_doppler_bins	- candidate bin count, capped at the full grid size. Use 0 for the full range defined by doppler_max and doppler_step, 1 for a known Doppler, or larger counts for partial uncertainty.

12.300.2.10 set_gnss_synchro()

```
void pcps_acquisition::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 106 of file pcps_acquisition.h.

12.300.2.11 set_local_code()

```
void pcps_acquisition::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

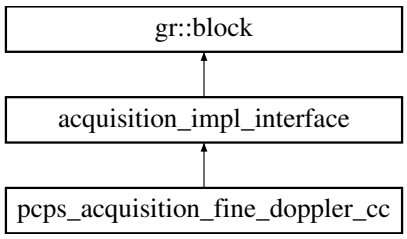
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_acquisition.h

12.301 pcps_acquisition_fine_doppler_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition.
`#include <pcps_acquisition_fine_doppler_cc.h>`
Inheritance diagram for pcps_acquisition_fine_doppler_cc:



Public Member Functions

- ~pcps_acquisition_fine_doppler_cc ()=default

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- unsigned int mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code) override

  Sets local code for PCPS acquisition algorithm.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (unsigned int channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_channel (uint32_t channel_id)=0
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_acquisition_fine_doppler_cc_sptr **pcps_make_acquisition_fine_doppler_cc** (const Acq_Conf &conf_)

12.301.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Definition at line 74 of file pcps_acquisition_fine_doppler_cc.h.

12.301.2 Constructor & Destructor Documentation

12.301.2.1 ~pcps_acquisition_fine_doppler_cc()

```
pcps_acquisition_fine_doppler_cc::~pcps_acquisition_fine_doppler_cc ()  [default]
```

Default destructor.

12.301.3 Member Function Documentation

12.301.3.1 general_work()

```
int pcps_acquisition_fine_doppler_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.301.3.2 mag()

```
unsigned int pcps_acquisition_fine_doppler_cc::mag () const  [inline], [override], [virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 95 of file pcps_acquisition_fine_doppler_cc.h.

12.301.3.3 set_active()

```
void pcps_acquisition_fine_doppler_cc::set_active (
            bool active)  [inline], [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 111 of file pcps_acquisition_fine_doppler_cc.h.

12.301.3.4 set_channel()

```
void pcps_acquisition_fine_doppler_cc::set_channel (
            unsigned int channel)  [inline], [override]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Definition at line 125 of file pcps_acquisition_fine_doppler_cc.h.

12.301.3.5 set_channel_fsm()

```
void pcps_acquisition_fine_doppler_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 134 of file pcps_acquisition_fine_doppler_cc.h.

12.301.3.6 set_gnss_synchro()

```
void pcps_acquisition_fine_doppler_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 87 of file pcps_acquisition_fine_doppler_cc.h.

12.301.3.7 set_local_code()

```
void pcps_acquisition_fine_doppler_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_acquisition_fine_doppler_cc.h

12.302 pcps_acquisition_fpga Class Reference

This class implements a Parallel Code Phase Search Acquisition that uses the FPGA.
`#include <pcps_acquisition_fpga.h>`

Public Member Functions

- ~pcps_acquisition_fpga ()=default

  Destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro)

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const

  Returns the maximum peak of grid search.
- void init ()

  Initializes acquisition algorithm.
- void set_local_code ()

  Sets local code for PCPS acquisition algorithm.
- void set_active (bool active)

Starts acquisition algorithm, turning from standby mode to active mode.

- void set_channel (uint32_t channel)

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm)

  Set channel fsm associated to this acquisition instance.
- void set_doppler_center (int32_t doppler_center)

  Set Doppler center frequency for the grid search. It will refresh the Doppler grid.
- void reset_acquisition ()

  This function triggers a HW reset of the FPGA PL.
- void stop_acquisition ()

  stop the acquisition and the other FPGA modules.

Friends

- pcps_acquisition_fpga_sptr **pcps_make_acquisition_fpga** (Acq_Conf_Fpga *conf, uint32_t acq_buff_num, std::vector< std::pair< uint32_t, uint32_t > > &downsampling_filter_specs, uint32_t &max_FFT_size)

12.302.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition that uses the FPGA.
Check An Open Source Galileo E1 Software Receiver, Algorithm 1, for a pseudocode description of this implementation.
Definition at line 58 of file pcps_acquisition_fpga.h.

12.302.2 Constructor & Destructor Documentation

12.302.2.1 ~pcps_acquisition_fpga()

```
pcps_acquisition_fpga::~pcps_acquisition_fpga ()  [default]
```

Destructor.

12.302.3 Member Function Documentation

12.302.3.1 init()

```
void pcps_acquisition_fpga::init ()
```

Initializes acquisition algorithm.

12.302.3.2 mag()

```
uint32_t pcps_acquisition_fpga::mag () const  [inline]
```

Returns the maximum peak of grid search.
Definition at line 79 of file pcps_acquisition_fpga.h.

12.302.3.3 reset_acquisition()

```
void pcps_acquisition_fpga::reset_acquisition ()
```

This function triggers a HW reset of the FPGA PL.

12.302.3.4 set_active()

```
void pcps_acquisition_fpga::set_active (
            bool active)
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

12.302.3.5 set_channel()

```
void pcps_acquisition_fpga::set_channel (
            uint32_t channel)  [inline]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Definition at line 105 of file pcps_acquisition_fpga.h.

12.302.3.6 set_channel_fsm()

```
void pcps_acquisition_fpga::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline]
```

Set channel fsm associated to this acquisition instance.
Definition at line 113 of file pcps_acquisition_fpga.h.

12.302.3.7 set_doppler_center()

```
void pcps_acquisition_fpga::set_doppler_center (
            int32_t doppler_center)
```

Set Doppler center frequency for the grid search. It will refresh the Doppler grid.

Parameters

doppler_center	- Frequency center of the search grid [Hz].

12.302.3.8 set_gnss_synchro()

```
void pcps_acquisition_fpga::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Definition at line 71 of file pcps_acquisition_fpga.h.

12.302.3.9 set_local_code()

```
void pcps_acquisition_fpga::set_local_code ()
```

Sets local code for PCPS acquisition algorithm.

12.302.3.10 stop_acquisition()

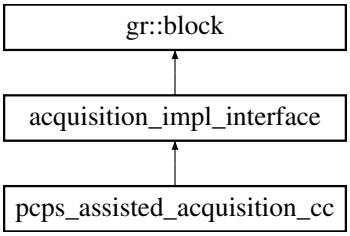
```
void pcps_acquisition_fpga::stop_acquisition ()
```

stop the acquisition and the other FPGA modules.
The documentation for this class was generated from the following file:

- pcps_acquisition_fpga.h

12.303 pcps_assisted_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition.
`#include <pcps_assisted_acquisition_cc.h>`
Inheritance diagram for pcps_assisted_acquisition_cc:



Public Member Functions

- ~pcps_assisted_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code) override

  Sets local code for PCPS acquisition algorithm.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_assisted_acquisition_cc_sptr **pcps_make_assisted_acquisition_cc** (const Acq_Conf &conf)

12.303.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Check An Open Source Galileo E1 Software Receiver, Algorithm 1, for a pseudocode description of this implementation.
Definition at line 68 of file pcps_assisted_acquisition_cc.h.

12.303.2 Constructor & Destructor Documentation

12.303.2.1 ~pcps_assisted_acquisition_cc()

```
pcps_assisted_acquisition_cc::~pcps_assisted_acquisition_cc ()
```
Default destructor.

12.303.3 Member Function Documentation

12.303.3.1 general_work()

```
int pcps_assisted_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```
Parallel Code Phase Search Acquisition signal processing.

12.303.3.2 mag()

```
uint32_t pcps_assisted_acquisition_cc::mag () const  [inline], [override], [virtual]
```
Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 89 of file pcps_assisted_acquisition_cc.h.

12.303.3.3 set_active()

```
void pcps_assisted_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```
Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 105 of file pcps_assisted_acquisition_cc.h.

12.303.3.4 set_channel()

```
void pcps_assisted_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```
Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 119 of file pcps_assisted_acquisition_cc.h.

12.303.3.5 set_channel_fsm()

```
void pcps_assisted_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```
Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 127 of file pcps_assisted_acquisition_cc.h.

12.303.3.6 set_gnss_synchro()

```
void pcps_assisted_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 81 of file pcps_assisted_acquisition_cc.h.

12.303.3.7 set_local_code()

```
void pcps_assisted_acquisition_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

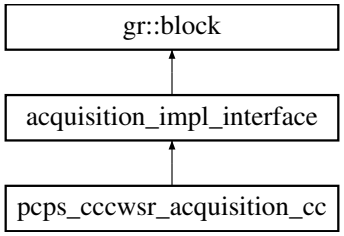
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_assisted_acquisition_cc.h

12.304 pcps_cccwsr_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme.
`#include <pcps_cccwsr_acquisition_cc.h>`
Inheritance diagram for pcps_cccwsr_acquisition_cc:



Public Member Functions

- ~pcps_cccwsr_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code_data, std::complex< float > *code_pilot) override

  Sets local code for CCCWSR acquisition algorithm.

- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Coherent Channel Combining With Sign Recovery Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *)

Friends

- pcps_cccwsr_acquisition_cc_sptr **pcps_cccwsr_make_acquisition_cc** (const Acq_Conf &conf)

12.304.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme.
Definition at line 55 of file pcps_cccwsr_acquisition_cc.h.

12.304.2 Constructor & Destructor Documentation

12.304.2.1 ~pcps_cccwsr_acquisition_cc()

```
pcps_cccwsr_acquisition_cc::~pcps_cccwsr_acquisition_cc ()
```
Default destructor.

12.304.3 Member Function Documentation

12.304.3.1 general_work()

```
int pcps_cccwsr_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```
Coherent Channel Combining With Sign Recovery Acquisition signal processing.

12.304.3.2 mag()

```
uint32_t pcps_cccwsr_acquisition_cc::mag () const  [inline], [override], [virtual]
```
Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 76 of file pcps_cccwsr_acquisition_cc.h.

12.304.3.3 set_active()

```
void pcps_cccwsr_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```
Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 93 of file pcps_cccwsr_acquisition_cc.h.

12.304.3.4 set_channel()

```
void pcps_cccwsr_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 107 of file pcps_cccwsr_acquisition_cc.h.

12.304.3.5 set_channel_fsm()

```
void pcps_cccwsr_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 115 of file pcps_cccwsr_acquisition_cc.h.

12.304.3.6 set_gnss_synchro()

```
void pcps_cccwsr_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 68 of file pcps_cccwsr_acquisition_cc.h.

12.304.3.7 set_local_code()

```
void pcps_cccwsr_acquisition_cc::set_local_code (
            std::complex< float > * code_data,
            std::complex< float > * code_pilot)  [override], [virtual]
```

Sets local code for CCCWSR acquisition algorithm.

Parameters

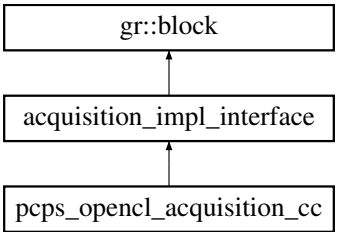
data_code	- Pointer to the data PRN code.
pilot_code	- Pointer to the pilot PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_cccwsr_acquisition_cc.h

12.305 pcps_opencl_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition.
`#include <pcps_opencl_acquisition_cc.h>`
Inheritance diagram for pcps_opencl_acquisition_cc:



Public Member Functions

- ~pcps_opencl_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code) override

  Sets local code for PCPS acquisition algorithm.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- bool opencl_ready () const
- void **acquisition_core_volk** ()
- void **acquisition_core_opencl** ()
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_opencl_acquisition_cc_sptr **pcps_make_opencl_acquisition_cc** (const Acq_Conf &conf, uint32_t max_dwells)

12.305.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Check An Open Source Galileo E1 Software Receiver, Algorithm 1, for a pseudocode description of this implementation.
Definition at line 75 of file pcps_opencl_acquisition_cc.h.

12.305.2 Constructor & Destructor Documentation

12.305.2.1 ~pcps_opencl_acquisition_cc()

```
pcps_opencl_acquisition_cc::~pcps_opencl_acquisition_cc ()
```
Default destructor.

12.305.3 Member Function Documentation

12.305.3.1 general_work()

```
int pcps_opencl_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```
Parallel Code Phase Search Acquisition signal processing.

12.305.3.2 mag()

```
uint32_t pcps_opencl_acquisition_cc::mag () const  [inline], [override], [virtual]
```
Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 96 of file pcps_opencl_acquisition_cc.h.

12.305.3.3 opencl_ready()

```
bool pcps_opencl_acquisition_cc::opencl_ready () const  [inline]
```
Definition at line 139 of file pcps_opencl_acquisition_cc.h.

12.305.3.4 set_active()

```
void pcps_opencl_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```
Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 112 of file pcps_opencl_acquisition_cc.h.

12.305.3.5 set_channel()

```
void pcps_opencl_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```
Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 126 of file pcps_opencl_acquisition_cc.h.

12.305.3.6 set_channel_fsm()

```
void pcps_opencl_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 134 of file pcps_opencl_acquisition_cc.h.

12.305.3.7 set_gnss_synchro()

```
void pcps_opencl_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 88 of file pcps_opencl_acquisition_cc.h.

12.305.3.8 set_local_code()

```
void pcps_opencl_acquisition_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

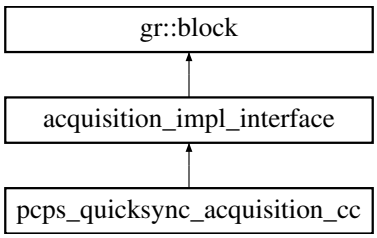
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_opencl_acquisition_cc.h

12.306 pcps_quicksync_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm.
`#include <pcps_quicksync_acquisition_cc.h>`
Inheritance diagram for pcps_quicksync_acquisition_cc:



Public Member Functions

- ~pcps_quicksync_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

Returns the maximum peak of grid search.

- void set_local_code (std::complex< float > *code) override

  Sets local code for PCPS acquisition algorithm.

- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.

- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.

- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.

- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_quicksync_acquisition_cc_sptr **pcps_quicksync_make_acquisition_cc** (const Acq_Conf &conf, uint32_t folding_factor, uint32_t max_dwells)

12.306.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm.
Check Faster GPS via the Sparse Fourier Transform, for details of its implementation and functionality.
Definition at line 73 of file pcps_quicksync_acquisition_cc.h.

12.306.2 Constructor & Destructor Documentation

12.306.2.1 ~pcps_quicksync_acquisition_cc()

```
pcps_quicksync_acquisition_cc::~pcps_quicksync_acquisition_cc ()
```

Default destructor.

12.306.3 Member Function Documentation

12.306.3.1 general_work()

```
int pcps_quicksync_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.306.3.2 mag()

```
uint32_t pcps_quicksync_acquisition_cc::mag () const  [inline], [override], [virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 94 of file pcps_quicksync_acquisition_cc.h.

12.306.3.3 set_active()

```
void pcps_quicksync_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 110 of file pcps_quicksync_acquisition_cc.h.

12.306.3.4 set_channel()

```
void pcps_quicksync_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 124 of file pcps_quicksync_acquisition_cc.h.

12.306.3.5 set_channel_fsm()

```
void pcps_quicksync_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 132 of file pcps_quicksync_acquisition_cc.h.

12.306.3.6 set_gnss_synchro()

```
void pcps_quicksync_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 86 of file pcps_quicksync_acquisition_cc.h.

12.306.3.7 set_local_code()

```
void pcps_quicksync_acquisition_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for PCPS acquisition algorithm.

Parameters

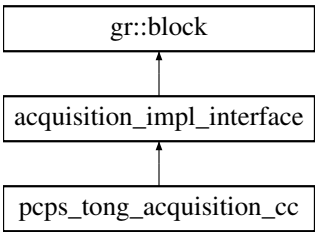
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_quicksync_acquisition_cc.h

12.307 pcps_tong_acquisition_cc Class Reference

This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.
`#include <pcps_tong_acquisition_cc.h>`
Inheritance diagram for pcps_tong_acquisition_cc:



Public Member Functions

- ~pcps_tong_acquisition_cc ()

  Default destructor.
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

  Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.
- uint32_t mag () const override

  Returns the maximum peak of grid search.
- void set_local_code (std::complex< float > *code) override

  Sets local code for TONG acquisition algorithm.
- void set_active (bool active) override

  Starts acquisition algorithm, turning from standby mode to active mode.
- void set_channel (uint32_t channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  Parallel Code Phase Search Acquisition signal processing.

Public Member Functions inherited from acquisition_impl_interface

- acquisition_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::sptr output_signature)
- virtual void set_local_code (std::complex< float > *, std::complex< float > *)

Friends

- pcps_tong_acquisition_cc_sptr **pcps_tong_make_acquisition_cc** (const Acq_Conf &conf, uint32_t tong_init_val, uint32_t tong_max_val, uint32_t tong_max_dwells)

12.307.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.
Definition at line 73 of file pcps_tong_acquisition_cc.h.

12.307.2 Constructor & Destructor Documentation

12.307.2.1 ~pcps_tong_acquisition_cc()

```
pcps_tong_acquisition_cc::~pcps_tong_acquisition_cc ()
```

Default destructor.

12.307.3 Member Function Documentation

12.307.3.1 general_work()

```
int pcps_tong_acquisition_cc::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

Parallel Code Phase Search Acquisition signal processing.

12.307.3.2 mag()

```
uint32_t pcps_tong_acquisition_cc::mag () const  [inline], [override], [virtual]
```

Returns the maximum peak of grid search.
Implements acquisition_impl_interface.
Definition at line 94 of file pcps_tong_acquisition_cc.h.

12.307.3.3 set_active()

```
void pcps_tong_acquisition_cc::set_active (
            bool active)  [inline], [override], [virtual]
```

Starts acquisition algorithm, turning from standby mode to active mode.

Parameters

active	- bool that activates/deactivates the block.

Implements acquisition_impl_interface.
Definition at line 110 of file pcps_tong_acquisition_cc.h.

12.307.3.4 set_channel()

```
void pcps_tong_acquisition_cc::set_channel (
            uint32_t channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.

Parameters

channel	- receiver channel.

Implements acquisition_impl_interface.
Definition at line 124 of file pcps_tong_acquisition_cc.h.

12.307.3.5 set_channel_fsm()

```
void pcps_tong_acquisition_cc::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements acquisition_impl_interface.
Definition at line 132 of file pcps_tong_acquisition_cc.h.

12.307.3.6 set_gnss_synchro()

```
void pcps_tong_acquisition_cc::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [inline], [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to exchange synchronization data between acquisition and tracking blocks.

Parameters

p_gnss_synchro	Satellite information shared by the processing blocks.

Implements acquisition_impl_interface.
Definition at line 86 of file pcps_tong_acquisition_cc.h.

12.307.3.7 set_local_code()

```
void pcps_tong_acquisition_cc::set_local_code (
            std::complex< float > * code)  [override], [virtual]
```

Sets local code for TONG acquisition algorithm.

Parameters

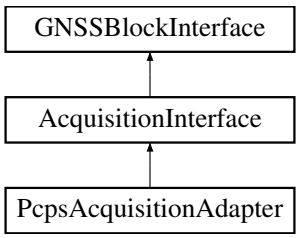
code	- Pointer to the PRN code.

Reimplemented from acquisition_impl_interface.
The documentation for this class was generated from the following file:

- pcps_tong_acquisition_cc.h

12.308 PcpsAcquisitionAdapter Class Reference

This class adapts a PCPS acquisition block to an AcquisitionInterface.
`#include <pcps_acquisition_adapter.h>`
Inheritance diagram for PcpsAcquisitionAdapter:



Public Member Functions

- **PcpsAcquisitionAdapter** (const ConfigurationInterface *configuration, const std::string &role, const std::↩ string &implementation, unsigned int in_streams, unsigned int out_streams, signal_flag sig_flag)
- std::string role () override
- std::string implementation () override
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.

- void set_channel (unsigned int channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- void set_doppler_center (int doppler_center) override

  Set Doppler center for the grid search.
- void set_doppler_num_bins (unsigned int doppler_num_bins) override

  Set the number of Doppler bins to search, centered on set_doppler_center() – see pcps_acquisition::set_doppler_num_bins().
- signed int mag () override

  Returns the maximum peak of grid search.
- void reset () override

  Restart acquisition algorithm.
- void stop_acquisition () override

  Stop running acquisition.
- void set_resampler_latency (uint32_t latency_samples) override

  Sets the resampler latency to account it in the acquisition code delay estimation.
- void set_local_code () override

  Sets local code.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.308.1 Detailed Description

This class adapts a PCPS acquisition block to an AcquisitionInterface.
Definition at line 44 of file pcps_acquisition_adapter.h.

12.308.2 Member Function Documentation

12.308.2.1 connect()

```
void PcpsAcquisitionAdapter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.308.2.2 disconnect()

```
void PcpsAcquisitionAdapter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.308.2.3 get_left_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapter::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.308.2.4 get_right_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapter::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.308.2.5 implementation()

```
std::string PcpsAcquisitionAdapter::implementation ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 62 of file pcps_acquisition_adapter.h.

12.308.2.6 item_size()

```
size_t PcpsAcquisitionAdapter::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 67 of file pcps_acquisition_adapter.h.

12.308.2.7 mag()

```
signed int PcpsAcquisitionAdapter::mag ()  [override], [virtual]
```

Returns the maximum peak of grid search.
Implements AcquisitionInterface.

12.308.2.8 reset()

```
void PcpsAcquisitionAdapter::reset ()  [override], [virtual]
```

Restart acquisition algorithm.
Implements AcquisitionInterface.

12.308.2.9 role()

```
std::string PcpsAcquisitionAdapter::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 57 of file pcps_acquisition_adapter.h.

12.308.2.10 set_channel()

```
void PcpsAcquisitionAdapter::set_channel (
            unsigned int channel)  [inline], [override], [virtual]
```

Set acquisition channel unique ID.
Implements AcquisitionInterface.
Definition at line 87 of file pcps_acquisition_adapter.h.

12.308.2.11 set_channel_fsm()

```
void PcpsAcquisitionAdapter::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [inline], [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements AcquisitionInterface.
Definition at line 95 of file pcps_acquisition_adapter.h.

12.308.2.12 set_doppler_center()

```
void PcpsAcquisitionAdapter::set_doppler_center (
            int doppler_center)  [override], [virtual]
```

Set Doppler center for the grid search.
Reimplemented from AcquisitionInterface.

12.308.2.13 set_doppler_num_bins()

```
void PcpsAcquisitionAdapter::set_doppler_num_bins (
            unsigned int doppler_num_bins)  [override], [virtual]
```

Set the number of Doppler bins to search, centered on set_doppler_center() – see pcps_acquisition::set_doppler_num_bins().
Reimplemented from AcquisitionInterface.

12.308.2.14 set_gnss_synchro()

```
void PcpsAcquisitionAdapter::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements AcquisitionInterface.

12.308.2.15 set_local_code()

```
void PcpsAcquisitionAdapter::set_local_code ()  [override], [virtual]
```

Sets local code.
Implements AcquisitionInterface.

12.308.2.16 set_resampler_latency()

```
void PcpsAcquisitionAdapter::set_resampler_latency (
            uint32_t latency_samples)  [override], [virtual]
```

Sets the resampler latency to account it in the acquisition code delay estimation.
Implements AcquisitionInterface.

12.308.2.17 stop_acquisition()

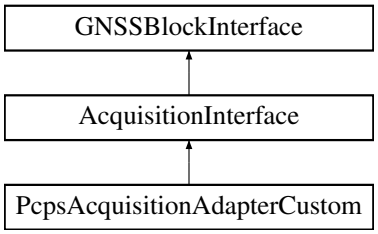
```
void PcpsAcquisitionAdapter::stop_acquisition ()  [override], [virtual]
```

Stop running acquisition.
Implements AcquisitionInterface.
The documentation for this class was generated from the following file:

- pcps_acquisition_adapter.h

12.309 PcpsAcquisitionAdapterCustom Class Reference

This class adapts a PCPS acquisition block to an AcquisitionInterface.
`#include <pcps_acquisition_adapter_custom.h>`
Inheritance diagram for PcpsAcquisitionAdapterCustom:



Public Member Functions

- **PcpsAcquisitionAdapterCustom** (const ConfigurationInterface *configuration, const std::string &role, const std::string &implementation, unsigned int in_streams, unsigned int out_streams, signal_flag sig_flag)
- std::string role () override
- std::string implementation () override
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.

- void set_channel (unsigned int channel) override

  Set acquisition channel unique ID.
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override

  Set channel fsm associated to this acquisition instance.
- signed int mag () override

  Returns the maximum peak of grid search.
- void reset () override

  Restart acquisition algorithm.
- void stop_acquisition () override

  Stop running acquisition.
- void set_resampler_latency (uint32_t) override
- void set_local_code () override

  Sets local code.

Public Member Functions inherited from AcquisitionInterface

- virtual void set_doppler_center (int)
- virtual void set_doppler_num_bins (unsigned int)

  Number of Doppler bins to search, centered on set_doppler_center(). doppler_num_bins == 0 searches the full configured Doppler range (computed from doppler_max/doppler_step – the one value a caller can't supply directly, since that count is implementation-private); any other value is the literal candidate bin count to search – 1 for an exactly-known Doppler (as when assisted by an already-tracked primary frequency or a visibility-aware prediction), or any N in between for a search with partial uncertainty.

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.309.1 Detailed Description

This class adapts a PCPS acquisition block to an AcquisitionInterface.
Definition at line 43 of file pcps_acquisition_adapter_custom.h.

12.309.2 Member Function Documentation

12.309.2.1 connect()

```
void PcpsAcquisitionAdapterCustom::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.309.2.2 disconnect()

```
void PcpsAcquisitionAdapterCustom::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.309.2.3 get_left_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapterCustom::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.309.2.4 get_right_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapterCustom::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.309.2.5 implementation()

```
std::string PcpsAcquisitionAdapterCustom::implementation ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 57 of file pcps_acquisition_adapter_custom.h.

12.309.2.6 item_size()

```
size_t PcpsAcquisitionAdapterCustom::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 59 of file pcps_acquisition_adapter_custom.h.

12.309.2.7 mag()

```
signed int PcpsAcquisitionAdapterCustom::mag ()  [override], [virtual]
```

Returns the maximum peak of grid search.
Implements AcquisitionInterface.

12.309.2.8 reset()

```
void PcpsAcquisitionAdapterCustom::reset ()  [override], [virtual]
```

Restart acquisition algorithm.
Implements AcquisitionInterface.

12.309.2.9 role()

```
std::string PcpsAcquisitionAdapterCustom::role ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 55 of file pcps_acquisition_adapter_custom.h.

12.309.2.10 set_channel()

```
void PcpsAcquisitionAdapterCustom::set_channel (
            unsigned int channel)  [override], [virtual]
```

Set acquisition channel unique ID.
Implements AcquisitionInterface.

12.309.2.11 set_channel_fsm()

```
void PcpsAcquisitionAdapterCustom::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [override], [virtual]
```

Set channel fsm associated to this acquisition instance.
Implements AcquisitionInterface.

12.309.2.12 set_gnss_synchro()

```
void PcpsAcquisitionAdapterCustom::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```

Set acquisition/tracking common Gnss_Synchro object pointer to efficiently exchange synchronization data between acquisition and tracking blocks.
Implements AcquisitionInterface.

12.309.2.13 set_local_code()

```
void PcpsAcquisitionAdapterCustom::set_local_code ()  [override], [virtual]
```

Sets local code.
Implements AcquisitionInterface.

12.309.2.14 set_resampler_latency()

```
void PcpsAcquisitionAdapterCustom::set_resampler_latency (
            uint32_t )  [inline], [override], [virtual]
```

Implements AcquisitionInterface.
Definition at line 98 of file pcps_acquisition_adapter_custom.h.

12.309.2.15 stop_acquisition()

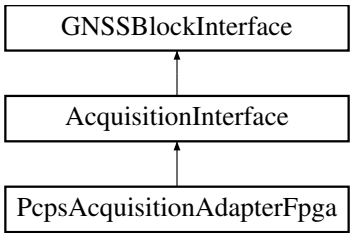
```
void PcpsAcquisitionAdapterCustom::stop_acquisition ()  [override], [virtual]
```

Stop running acquisition.
Implements AcquisitionInterface.
The documentation for this class was generated from the following file:

- pcps_acquisition_adapter_custom.h

12.310 PcpsAcquisitionAdapterFpga Class Reference

Inheritance diagram for PcpsAcquisitionAdapterFpga:



Public Member Functions

- **PcpsAcquisitionAdapterFpga** (const ConfigurationInterface *configuration, const std::string &role, const std::string &implementation, unsigned int in_streams, unsigned int out_streams, signal_flag sig_flag)
- std::string role () override
- std::string implementation () override
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- signed int mag () override
- void set_gnss_synchro (Gnss_Synchro *p_gnss_synchro) override
- void set_channel (unsigned int channel) override
- void set_channel_fsm (std::weak_ptr< ChannelFsm > channel_fsm) override
- void set_doppler_center (int doppler_center) override
- void reset () override
- void stop_acquisition () override
- void set_resampler_latency (uint32_t latency_samples __attribute__((unused))) override
- void set_local_code () override

Public Member Functions inherited from AcquisitionInterface

- virtual void set_doppler_num_bins (unsigned int)

  Number of Doppler bins to search, centered on set_doppler_center(). doppler_num_bins == 0 searches the full configured Doppler range (computed from doppler_max/doppler_step – the one value a caller can't supply directly, since that count is implementation-private); any other value is the literal candidate bin count to search – 1 for an exactly-known Doppler (as when assisted by an already-tracked primary frequency or a visibility-aware prediction), or any N in between for a search with partial uncertainty.
- virtual void set_resampler_latency (uint32_t latency_samples)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.310.1 Detailed Description

Definition at line 35 of file pcps_acquisition_adapter_fpga.h.

12.310.2 Member Function Documentation

12.310.2.1 connect()

```
void PcpsAcquisitionAdapterFpga::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.310.2.2 disconnect()

```
void PcpsAcquisitionAdapterFpga::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.310.2.3 get_left_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapterFpga::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.310.2.4 get_right_block()

```
gr::basic_block_sptr PcpsAcquisitionAdapterFpga::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.310.2.5 implementation()

```
std::string PcpsAcquisitionAdapterFpga::implementation ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 49 of file pcps_acquisition_adapter_fpga.h.

12.310.2.6 item_size()

```
size_t PcpsAcquisitionAdapterFpga::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 50 of file pcps_acquisition_adapter_fpga.h.

12.310.2.7 mag()

```
signed int PcpsAcquisitionAdapterFpga::mag ()  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.8 reset()

```
void PcpsAcquisitionAdapterFpga::reset ()  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.9 role()

```
std::string PcpsAcquisitionAdapterFpga::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 48 of file pcps_acquisition_adapter_fpga.h.

12.310.2.10 set_channel()

```
void PcpsAcquisitionAdapterFpga::set_channel (
            unsigned int channel)  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.11 set_channel_fsm()

```
void PcpsAcquisitionAdapterFpga::set_channel_fsm (
            std::weak_ptr< ChannelFsm > channel_fsm)  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.12 set_doppler_center()

```
void PcpsAcquisitionAdapterFpga::set_doppler_center (
            int doppler_center)  [override], [virtual]
```
Reimplemented from AcquisitionInterface.

12.310.2.13 set_gnss_synchro()

```
void PcpsAcquisitionAdapterFpga::set_gnss_synchro (
            Gnss_Synchro * p_gnss_synchro)  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.14 set_local_code()

```
void PcpsAcquisitionAdapterFpga::set_local_code ()  [override], [virtual]
```
Implements AcquisitionInterface.

12.310.2.15 set_resampler_latency()

```
void PcpsAcquisitionAdapterFpga::set_resampler_latency (
            uint32_t latency_samples __attribute__(unused))  [inline], [override]
```
Definition at line 64 of file pcps_acquisition_adapter_fpga.h.

12.310.2.16 stop_acquisition()

```
void PcpsAcquisitionAdapterFpga::stop_acquisition ()  [override], [virtual]
```
Implements AcquisitionInterface.
The documentation for this class was generated from the following file:

- pcps_acquisition_adapter_fpga.h

12.311 pcv_t Struct Reference

Public Attributes

- int sat
- char type [MAXANT]
- char code [MAXANT]
- gtime_t ts
- gtime_t te
- double off [NFREQ][3]
- double var [NFREQ][19]

12.311.1 Detailed Description

Definition at line 428 of file rtklib.h.

12.311.2 Member Data Documentation

12.311.2.1 code

```
char pcv_t::code[MAXANT]
```

Definition at line 432 of file rtklib.h.

12.311.2.2 off

```
double pcv_t::off[NFREQ][3]
```

Definition at line 434 of file rtklib.h.

12.311.2.3 sat

```
int pcv_t::sat
```

Definition at line 430 of file rtklib.h.

12.311.2.4 te

```
gtime_t pcv_t::te
```

Definition at line 433 of file rtklib.h.

12.311.2.5 ts

```
gtime_t pcv_t::ts
```

Definition at line 433 of file rtklib.h.

12.311.2.6 type

```
char pcv_t::type[MAXANT]
```

Definition at line 431 of file rtklib.h.

12.311.2.7 var

```
double pcv_t::var[NFREQ][19]
```

Definition at line 435 of file rtklib.h.

The documentation for this struct was generated from the following file:

- rtklib.h

12.312 pcvs_t Struct Reference

Public Attributes

- int n
- int nmax
- pcv_t * pcv

12.312.1 Detailed Description

Definition at line 440 of file rtklib.h.

12.312.2 Member Data Documentation

12.312.2.1 n

```
int pcvs_t::n
```

Definition at line 442 of file rtklib.h.

12.312.2.2 nmax

```
int pcvs_t::nmax
```

Definition at line 442 of file rtklib.h.

12.312.2.3 pcv

```
pcv_t* pcvs_t::pcv
```

Definition at line 443 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.313 peph_t Struct Reference

Public Attributes

- gtime_t time
- int index
- double pos [MAXSAT][4]
- float std [MAXSAT][4]
- double vel [MAXSAT][4]
- float vst [MAXSAT][4]
- float cov [MAXSAT][3]
- float vco [MAXSAT][3]

12.313.1 Detailed Description

Definition at line 514 of file rtklib.h.

12.313.2 Member Data Documentation

12.313.2.1 cov

```
float peph_t::cov[MAXSAT][3]
```

Definition at line 522 of file rtklib.h.

12.313.2.2 index

```
int peph_t::index
```

Definition at line 517 of file rtklib.h.

12.313.2.3 pos

`double peph_t::pos[MAXSAT][4]`
Definition at line 518 of file rtklib.h.

12.313.2.4 std

`float peph_t::std[MAXSAT][4]`
Definition at line 519 of file rtklib.h.

12.313.2.5 time

`gtime_t peph_t::time`
Definition at line 516 of file rtklib.h.

12.313.2.6 vco

`float peph_t::vco[MAXSAT][3]`
Definition at line 523 of file rtklib.h.

12.313.2.7 vel

`double peph_t::vel[MAXSAT][4]`
Definition at line 520 of file rtklib.h.

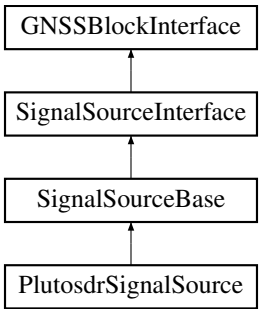
12.313.2.8 vst

`float peph_t::vst[MAXSAT][4]`
Definition at line 521 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.314 PlutosdrSignalSource Class Reference

Inheritance diagram for PlutosdrSignalSource:



Public Member Functions

- **PlutosdrSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in↩ _stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.314.1 Detailed Description

Definition at line 49 of file plutosdr_signal_source.h.

12.314.2 Member Function Documentation

12.314.2.1 connect()

```
void PlutosdrSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.314.2.2 disconnect()

```
void PlutosdrSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.314.2.3 get_left_block()

```
gr::basic_block_sptr PlutosdrSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.314.2.4 get_right_block()

```
gr::basic_block_sptr PlutosdrSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.314.2.5 item_size()

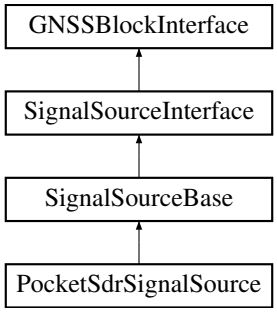
```
size_t PlutosdrSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 58 of file plutosdr_signal_source.h.
The documentation for this class was generated from the following file:

- plutosdr_signal_source.h

12.315 PocketSdrSignalSource Class Reference

This class instantiates the gr-pocketsdr signal source for the Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends.
`#include <pocket_sdr_signal_source.h>`
Inheritance diagram for PocketSdrSignalSource:



Public Member Functions

- **PocketSdrSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.315.1 Detailed Description

This class instantiates the gr-pocketsdr signal source for the Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends.
Definition at line 42 of file pocket_sdr_signal_source.h.

12.315.2 Member Function Documentation

12.315.2.1 connect()

```
void PocketSdrSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.315.2.2 disconnect()

```
void PocketSdrSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.315.2.3 get_left_block()

```
gr::basic_block_sptr PocketSdrSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.315.2.4 get_right_block()

```
gr::basic_block_sptr PocketSdrSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.315.2.5 item_size()

```
size_t PocketSdrSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 51 of file pocket_sdr_signal_source.h.
The documentation for this class was generated from the following file:

- pocket_sdr_signal_source.h

12.316 pppcorr_t Struct Reference

Public Attributes

- int nsta
- char stas [MAXSTA][8]
- double rr [MAXSTA][3]
- int ns [MAXSTA]
- int nsmax [MAXSTA]
- int nt [MAXSTA]
- int ntmax [MAXSTA]
- stec_t * stec [MAXSTA]
- trop_t * trop [MAXSTA]

12.316.1 Detailed Description

Definition at line 776 of file rtklib.h.

12.316.2 Member Data Documentation

12.316.2.1 ns

```
int pppcorr_t::ns[MAXSTA]
```

Definition at line 781 of file rtklib.h.

12.316.2.2 nsmax

```
int pppcorr_t::nsmax[MAXSTA]
```
Definition at line 781 of file rtklib.h.

12.316.2.3 nsta

```
int pppcorr_t::nsta
```
Definition at line 778 of file rtklib.h.

12.316.2.4 nt

```
int pppcorr_t::nt[MAXSTA]
```
Definition at line 782 of file rtklib.h.

12.316.2.5 ntmax

```
int pppcorr_t::ntmax[MAXSTA]
```
Definition at line 782 of file rtklib.h.

12.316.2.6 rr

```
double pppcorr_t::rr[MAXSTA][3]
```
Definition at line 780 of file rtklib.h.

12.316.2.7 stas

```
char pppcorr_t::stas[MAXSTA][8]
```
Definition at line 779 of file rtklib.h.

12.316.2.8 stec

```
stec_t* pppcorr_t::stec[MAXSTA]
```
Definition at line 783 of file rtklib.h.

12.316.2.9 trop

```
trop_t* pppcorr_t::trop[MAXSTA]
```
Definition at line 784 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.317 pps_tcp_rx Class Reference

Public Member Functions

- void **receive_pps** (const std::string &ip_address, int port)
- bool **send_cmd** (std::string cmd) const
- void **set_pps_samplestamp_queue** (std::shared_ptr< Concurrent_Queue< PpsSamplestamp > > queue)

Public Attributes

- volatile bool is_connected {false}

12.317.1 Detailed Description

Definition at line 27 of file ppstcprx.h.

12.317.2 Member Data Documentation

12.317.2.1 is_connected

```
volatile bool pps_tcp_rx::is_connected {false}
```
Definition at line 34 of file ppstcprx.h.
The documentation for this class was generated from the following file:

- ppstcprx.h

12.318 PpsSamplestamp Class Reference

Public Attributes

- uint64_t samplestamp = 0
- uint32_t overflow_reg = 0

12.318.1 Detailed Description

Definition at line 21 of file pps_samplestamp.h.

12.318.2 Member Data Documentation

12.318.2.1 overflow_reg

```
uint32_t PpsSamplestamp::overflow_reg = 0
```
Definition at line 25 of file pps_samplestamp.h.

12.318.2.2 samplestamp

```
uint64_t PpsSamplestamp::samplestamp = 0
```
Definition at line 24 of file pps_samplestamp.h.
The documentation for this class was generated from the following file:

- pps_samplestamp.h

12.319 prcopt_t Struct Reference

Public Attributes

- int mode
- int soltype
- int nf
- int navsys
- double elmin
- snrmask_t snrmask
- int sateph
- int modear
- int glomodear
- int bdsmodear
- int maxout
- int minlock
- int minfix
- int armaxiter
- int ionoopt
- int tropopt
- int dynamics
- int tidecorr

- int niter
- int codesmooth
- int intpref
- int sbascorr
- int sbassatsel
- int rovpos
- int refpos
- double eratio [NFREQ]
- double err [8]
- double std [3]
- double prn [6]
- double sclkstab
- double thresar [8]
- double elmaskar
- double elmaskhold
- double thresslip
- double thresdop
- double maxtdiff
- double maxinno [2]
- double maxgdop
- double baseline [2]
- double ru [3]
- double rb [3]
- char anttype [2][MAXANT]
- double antdel [2][3]
- pcv_t pcvr [2]
- unsigned char exsats [MAXSAT]
- int maxaveep
- int initrst
- int outsingle
- char rnxopt [2][256]
- int posopt [6]
- int syncsol
- double odisp [2][6 ∗11]
- exterr_t exterr
- int freqopt
- char pppopt [256]
- bool bancroft_init
- bool estqzsisb
- int arfilter
- int minfixsats
- int minholdsats
- int mindropsats
- double varholdamb
- double armaxposvar

12.319.1 Detailed Description

Definition at line 982 of file rtklib.h.

12.319.2 Member Data Documentation

12.319.2.1 antdel

```
double prcopt_t::antdel[2][3]
```

Definition at line 1034 of file rtklib.h.

12.319.2.2 anttype

```
char prcopt_t::anttype[2][MAXANT]
```
Definition at line 1033 of file rtklib.h.

12.319.2.3 arfilter

```
int prcopt_t::arfilter
```
Definition at line 1058 of file rtklib.h.

12.319.2.4 armaxiter

```
int prcopt_t::armaxiter
```
Definition at line 997 of file rtklib.h.

12.319.2.5 armaxposvar

```
double prcopt_t::armaxposvar
```
Definition at line 1063 of file rtklib.h.

12.319.2.6 bancroft_init

```
bool prcopt_t::bancroft_init
```
Definition at line 1047 of file rtklib.h.

12.319.2.7 baseline

```
double prcopt_t::baseline[2]
```
Definition at line 1030 of file rtklib.h.

12.319.2.8 bdsmodear

```
int prcopt_t::bdsmodear
```
Definition at line 993 of file rtklib.h.

12.319.2.9 codesmooth

```
int prcopt_t::codesmooth
```
Definition at line 1003 of file rtklib.h.

12.319.2.10 dynamics

```
int prcopt_t::dynamics
```
Definition at line 1000 of file rtklib.h.

12.319.2.11 elmaskar

```
double prcopt_t::elmaskar
```
Definition at line 1023 of file rtklib.h.

12.319.2.12 elmaskhold

```
double prcopt_t::elmaskhold
```
Definition at line 1024 of file rtklib.h.

12.319.2.13 elmin

```
double prcopt_t::elmin
```
Definition at line 988 of file rtklib.h.

12.319.2.14 eratio

```
double prcopt_t::eratio[NFREQ]
```
Definition at line 1011 of file rtklib.h.

12.319.2.15 err

```
double prcopt_t::err[8]
```
Definition at line 1012 of file rtklib.h.

12.319.2.16 estqzsisb

```
bool prcopt_t::estqzsisb
```
Definition at line 1048 of file rtklib.h.

12.319.2.17 exsats

```
unsigned char prcopt_t::exsats[MAXSAT]
```
Definition at line 1036 of file rtklib.h.

12.319.2.18 exterr

```
exterr_t prcopt_t::exterr
```
Definition at line 1044 of file rtklib.h.

12.319.2.19 freqopt

```
int prcopt_t::freqopt
```
Definition at line 1045 of file rtklib.h.

12.319.2.20 glomodear

```
int prcopt_t::glomodear
```
Definition at line 992 of file rtklib.h.

12.319.2.21 initrst

```
int prcopt_t::initrst
```
Definition at line 1038 of file rtklib.h.

12.319.2.22 intpref

```
int prcopt_t::intpref
```
Definition at line 1004 of file rtklib.h.

12.319.2.23 ionoopt

```
int prcopt_t::ionoopt
```
Definition at line 998 of file rtklib.h.

12.319.2.24 maxaveep

```
int prcopt_t::maxaveep
```
Definition at line 1037 of file rtklib.h.

12.319.2.25 maxgdop

```
double prcopt_t::maxgdop
```
Definition at line 1029 of file rtklib.h.

12.319.2.26 maxinno

```
double prcopt_t::maxinno[2]
```
Definition at line 1028 of file rtklib.h.

12.319.2.27 maxout

```
int prcopt_t::maxout
```
Definition at line 994 of file rtklib.h.

12.319.2.28 maxtdiff

```
double prcopt_t::maxtdiff
```
Definition at line 1027 of file rtklib.h.

12.319.2.29 mindropsats

```
int prcopt_t::mindropsats
```
Definition at line 1061 of file rtklib.h.

12.319.2.30 minfix

```
int prcopt_t::minfix
```
Definition at line 996 of file rtklib.h.

12.319.2.31 minfixsats

```
int prcopt_t::minfixsats
```
Definition at line 1059 of file rtklib.h.

12.319.2.32 minholdsats

```
int prcopt_t::minholdsats
```
Definition at line 1060 of file rtklib.h.

12.319.2.33 minlock

```
int prcopt_t::minlock
```
Definition at line 995 of file rtklib.h.

12.319.2.34 mode

```
int prcopt_t::mode
```
Definition at line 984 of file rtklib.h.

12.319.2.35 modear

```
int prcopt_t::modear
```
Definition at line 991 of file rtklib.h.

12.319.2.36 navsys

```
int prcopt_t::navsys
```
Definition at line 987 of file rtklib.h.

12.319.2.37 nf

```
int prcopt_t::nf
```
Definition at line 986 of file rtklib.h.

12.319.2.38 niter

```
int prcopt_t::niter
```
Definition at line 1002 of file rtklib.h.

12.319.2.39 odisp

```
double prcopt_t::odisp[2][6 *11]
```
Definition at line 1043 of file rtklib.h.

12.319.2.40 outsingle

```
int prcopt_t::outsingle
```
Definition at line 1039 of file rtklib.h.

12.319.2.41 pcvr

```
pcv_t prcopt_t::pcvr[2]
```
Definition at line 1035 of file rtklib.h.

12.319.2.42 posopt

```
int prcopt_t::posopt[6]
```
Definition at line 1041 of file rtklib.h.

12.319.2.43 pppopt

```
char prcopt_t::pppopt[256]
```
Definition at line 1046 of file rtklib.h.

12.319.2.44 prn

```
double prcopt_t::prn[6]
```
Definition at line 1020 of file rtklib.h.

12.319.2.45 rb

```
double prcopt_t::rb[3]
```
Definition at line 1032 of file rtklib.h.

12.319.2.46 refpos

```
int prcopt_t::refpos
```
Definition at line 1008 of file rtklib.h.

12.319.2.47 rnxopt

```
char prcopt_t::rnxopt[2][256]
```
Definition at line 1040 of file rtklib.h.

12.319.2.48 rovpos

```
int prcopt_t::rovpos
```
Definition at line 1007 of file rtklib.h.

12.319.2.49 ru

```
double prcopt_t::ru[3]
```
Definition at line 1031 of file rtklib.h.

12.319.2.50 sateph

`int prcopt_t::sateph`
Definition at line 990 of file rtklib.h.

12.319.2.51 sbascorr

`int prcopt_t::sbascorr`
Definition at line 1005 of file rtklib.h.

12.319.2.52 sbassatsel

`int prcopt_t::sbassatsel`
Definition at line 1006 of file rtklib.h.

12.319.2.53 sclkstab

`double prcopt_t::sclkstab`
Definition at line 1021 of file rtklib.h.

12.319.2.54 snrmask

`snrmask_t prcopt_t::snrmask`
Definition at line 989 of file rtklib.h.

12.319.2.55 soltype

`int prcopt_t::soltype`
Definition at line 985 of file rtklib.h.

12.319.2.56 std

`double prcopt_t::std[3]`
Definition at line 1019 of file rtklib.h.

12.319.2.57 syncsol

`int prcopt_t::syncsol`
Definition at line 1042 of file rtklib.h.

12.319.2.58 thresar

`double prcopt_t::thresar[8]`
Definition at line 1022 of file rtklib.h.

12.319.2.59 thresdop

`double prcopt_t::thresdop`
Definition at line 1026 of file rtklib.h.

12.319.2.60 thresslip

`double prcopt_t::thresslip`
Definition at line 1025 of file rtklib.h.

12.319.2.61 tidecorr

`int prcopt_t::tidecorr`
Definition at line 1001 of file rtklib.h.

12.319.2.62 tropopt

```
int prcopt_t::tropopt
```
Definition at line 999 of file rtklib.h.

12.319.2.63 varholdamb

```
double prcopt_t::varholdamb
```
Definition at line 1062 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.320 Protobuf_Cleanup_Manager Class Reference

This class implements the Protocol Buffers library shared memory cleanup manager.
`#include <protobuf_cleanup_manager.h>`

Public Member Functions

- **Protobuf_Cleanup_Manager** (const Protobuf_Cleanup_Manager &)=delete
- Protobuf_Cleanup_Manager & **operator=** (const Protobuf_Cleanup_Manager &)=delete
- **Protobuf_Cleanup_Manager** (Protobuf_Cleanup_Manager &&)=delete
- Protobuf_Cleanup_Manager & **operator=** (Protobuf_Cleanup_Manager &&)=delete

Static Public Member Functions

- static Protobuf_Cleanup_Manager & get ()

12.320.1 Detailed Description

This class implements the Protocol Buffers library shared memory cleanup manager.
google::protobuf::ShutdownProtobufLibrary() must be called only once, at the end of the program execution: it is not safe to use any other part of the library after that call. Calling it earlier (e.g., from the destructor of a class that uses protobuf) results in use-after-free memory corruption and random crashes. Call Protobuf_Cleanup_Manager::get() in the constructor of every class that uses the protobuf library, so that google::protobuf::ShutdownProtobufLibrary() is called at program shutdown, when the singleton is destroyed.
Definition at line 39 of file protobuf_cleanup_manager.h.

12.320.2 Member Function Documentation

12.320.2.1 get()

```
Protobuf_Cleanup_Manager & Protobuf_Cleanup_Manager::get ()  [inline], [static]
```
Definition at line 42 of file protobuf_cleanup_manager.h.
The documentation for this class was generated from the following file:

- protobuf_cleanup_manager.h

12.321 Osnma_Merkle_Tree_Material::PublicKeyEntry Struct Reference

Public Attributes

- bool pkid_valid {false}
- uint8_t pkid {0}
- uint32_t leaf_index {0}
- uint32_t length_bits {0}
- std::string key_type
- std::string point

12.321.1 Detailed Description

Definition at line 44 of file osnma_crypto_material.h.

12.321.2 Member Data Documentation

12.321.2.1 key_type

```
std::string Osnma_Merkle_Tree_Material::PublicKeyEntry::key_type
```
Definition at line 50 of file osnma_crypto_material.h.

12.321.2.2 leaf_index

```
uint32_t Osnma_Merkle_Tree_Material::PublicKeyEntry::leaf_index {0}
```
Definition at line 48 of file osnma_crypto_material.h.

12.321.2.3 length_bits

```
uint32_t Osnma_Merkle_Tree_Material::PublicKeyEntry::length_bits {0}
```
Definition at line 49 of file osnma_crypto_material.h.

12.321.2.4 pkid

```
uint8_t Osnma_Merkle_Tree_Material::PublicKeyEntry::pkid {0}
```
Definition at line 47 of file osnma_crypto_material.h.

12.321.2.5 pkid_valid

```
bool Osnma_Merkle_Tree_Material::PublicKeyEntry::pkid_valid {false}
```
Definition at line 46 of file osnma_crypto_material.h.

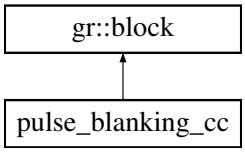
12.321.2.6 point

```
std::string Osnma_Merkle_Tree_Material::PublicKeyEntry::point
```
Definition at line 51 of file osnma_crypto_material.h.
The documentation for this struct was generated from the following file:

- osnma_crypto_material.h

12.322 pulse_blanking_cc Class Reference

Inheritance diagram for pulse_blanking_cc:



Public Member Functions

- int **general_work** (int noutput_items __attribute__((unused)), gr_vector_int &ninput_items __attribute_↩ _((unused)), gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- pulse_blanking_cc_sptr **make_pulse_blanking_cc** (float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)

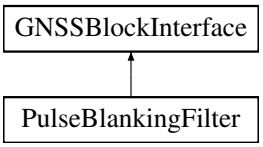
12.322.1 Detailed Description

Definition at line 44 of file pulse_blanking_cc.h.
The documentation for this class was generated from the following file:

- pulse_blanking_cc.h

12.323 PulseBlankingFilter Class Reference

Inheritance diagram for PulseBlankingFilter:



Public Member Functions

- **PulseBlankingFilter** (const ConfigurationInterface *configuration, std::string role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "Pulse_Blanking_Filter".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.323.1 Detailed Description

Definition at line 39 of file pulse_blanking_filter.h.

12.323.2 Member Function Documentation

12.323.2.1 connect()

```
void PulseBlankingFilter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.323.2.2 disconnect()

```
void PulseBlankingFilter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.323.2.3 get_left_block()

```
gr::basic_block_sptr PulseBlankingFilter::get_left_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.323.2.4 get_right_block()

```
gr::basic_block_sptr PulseBlankingFilter::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.323.2.5 implementation()

```
std::string PulseBlankingFilter::implementation ()  [inline], [override], [virtual]
```
Returns "Pulse_Blanking_Filter".
Implements GNSSBlockInterface.
Definition at line 54 of file pulse_blanking_filter.h.

12.323.2.6 item_size()

```
size_t PulseBlankingFilter::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 59 of file pulse_blanking_filter.h.

12.323.2.7 role()

```
std::string PulseBlankingFilter::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 48 of file pulse_blanking_filter.h.
The documentation for this class was generated from the following file:

- pulse_blanking_filter.h

12.324 Pvt_Conf Class Reference

Public Attributes

- std::map< int, int > rtcm_msg_rate_ms
- std::string rinex_name = std::string("-")
- std::string dump_filename
- std::string nmea_dump_filename
- std::string nmea_dump_devname
- std::string rtcm_dump_devname
- std::string an_dump_devname
- std::string output_path = std::string(".")
- std::string rinex_output_path = std::string(".")
- std::string gpx_output_path = std::string(".")
- std::string geojson_output_path = std::string(".")
- std::string nmea_output_file_path = std::string(".")
- std::string kml_output_path = std::string(".")
- std::string xml_output_path = std::string(".")
- std::string rtcm_output_file_path = std::string(".")
- std::string has_output_file_path = std::string(".")
- std::string udp_addresses
- std::string udp_ports
- std::string udp_eph_addresses
- std::string log_source_timetag_file
- std::string ntrip_caster_address
- std::string ntrip_mountpoint

- Secure_String ntrip_username
- Secure_String ntrip_password
- Secure_String ntrip_password_env
- uint32_t signal_enabled_flags = 0
- uint32_t observable_interval_ms = 20
- int32_t output_rate_ms = 0
- int32_t display_rate_ms = 0
- int32_t kml_rate_ms = 20
- int32_t gpx_rate_ms = 20
- int32_t geojson_rate_ms = 20
- int32_t nmea_rate_ms = 20
- int32_t rinex_version = 0
- int32_t rinexobs_rate_ms = 0
- int32_t an_rate_ms = 20
- int32_t max_obs_block_rx_clock_offset_ms = 40
- int32_t ref_gps_week = 0
- int udp_eph_port = 0
- int rtk_trace_level = 0
- int32_t ntrip_timeout_ms = 10000
- int32_t ntrip_reconnect_interval_ms = 10000
- int32_t ntrip_version = 2
- int32_t ntrip_gga_period_ms = 10000
- uint16_t rtcm_tcp_port = 0
- uint16_t rtcm_station_id = 0
- uint16_t ntrip_port = 2101
- uint16_t ntrip_station_id = 0
- double ntrip_max_correction_age_s = 5.0
- bool flag_nmea_tty_port = false
- bool flag_rtcm_server = false
- bool flag_rtcm_tty_port = false
- bool output_enabled = true
- bool rinex_output_enabled = true
- bool gpx_output_enabled = true
- bool geojson_output_enabled = true
- bool nmea_output_file_enabled = true
- bool an_output_enabled = false
- bool kml_output_enabled = true
- bool xml_output_enabled = true
- bool rtcm_output_file_enabled = true
- bool monitor_enabled = false
- bool monitor_ephemeris_enabled = false
- bool protobuf_enabled = true
- bool enable_rx_clock_correction = true
- bool show_local_time_zone = false
- bool dump = false
- bool dump_mat = true
- bool log_source_timetag = false
- bool ntrip_client_enabled = false
- bool ntrip_fallback_to_single = true
- bool ntrip_tls_enabled = false
- bool ntrip_send_gga = true
- bool use_e6_for_pvt = true
- bool use_has_corrections = true
- bool use_unhealthy_sats = false
- bool glonass_strict_health = true

- bool osnma_strict = false
- bool enable_pvt_kf = false
- double measures_ecef_pos_sd_m = 1.0
- double measures_ecef_vel_sd_ms = 0.1
- double system_ecef_pos_sd_m = 0.01
- double system_ecef_vel_sd_ms = 0.001
- bool kf_use_imu_vel = false

12.324.1 Detailed Description

Definition at line 31 of file pvt_conf.h.

12.324.2 Member Data Documentation

12.324.2.1 an_dump_devname

```
std::string Pvt_Conf::an_dump_devname
```

Definition at line 41 of file pvt_conf.h.

12.324.2.2 an_output_enabled

```
bool Pvt_Conf::an_output_enabled = false
```

Definition at line 101 of file pvt_conf.h.

12.324.2.3 an_rate_ms

```
int32_t Pvt_Conf::an_rate_ms = 20
```

Definition at line 76 of file pvt_conf.h.

12.324.2.4 display_rate_ms

```
int32_t Pvt_Conf::display_rate_ms = 0
```

Definition at line 69 of file pvt_conf.h.

12.324.2.5 dump

```
bool Pvt_Conf::dump = false
```

Definition at line 110 of file pvt_conf.h.

12.324.2.6 dump_filename

```
std::string Pvt_Conf::dump_filename
```

Definition at line 37 of file pvt_conf.h.

12.324.2.7 dump_mat

```
bool Pvt_Conf::dump_mat = true
```

Definition at line 111 of file pvt_conf.h.

12.324.2.8 enable_pvt_kf

```
bool Pvt_Conf::enable_pvt_kf = false
```

Definition at line 124 of file pvt_conf.h.

12.324.2.9 enable_rx_clock_correction

```
bool Pvt_Conf::enable_rx_clock_correction = true
```

Definition at line 108 of file pvt_conf.h.

12.324.2.10 flag_nmea_tty_port

```
bool Pvt_Conf::flag_nmea_tty_port = false
```
Definition at line 93 of file pvt_conf.h.

12.324.2.11 flag_rtcm_server

```
bool Pvt_Conf::flag_rtcm_server = false
```
Definition at line 94 of file pvt_conf.h.

12.324.2.12 flag_rtcm_tty_port

```
bool Pvt_Conf::flag_rtcm_tty_port = false
```
Definition at line 95 of file pvt_conf.h.

12.324.2.13 geojson_output_enabled

```
bool Pvt_Conf::geojson_output_enabled = true
```
Definition at line 99 of file pvt_conf.h.

12.324.2.14 geojson_output_path

```
std::string Pvt_Conf::geojson_output_path = std::string(".")
```
Definition at line 45 of file pvt_conf.h.

12.324.2.15 geojson_rate_ms

```
int32_t Pvt_Conf::geojson_rate_ms = 20
```
Definition at line 72 of file pvt_conf.h.

12.324.2.16 glonass_strict_health

```
bool Pvt_Conf::glonass_strict_health = true
```
Definition at line 120 of file pvt_conf.h.

12.324.2.17 gpx_output_enabled

```
bool Pvt_Conf::gpx_output_enabled = true
```
Definition at line 98 of file pvt_conf.h.

12.324.2.18 gpx_output_path

```
std::string Pvt_Conf::gpx_output_path = std::string(".")
```
Definition at line 44 of file pvt_conf.h.

12.324.2.19 gpx_rate_ms

```
int32_t Pvt_Conf::gpx_rate_ms = 20
```
Definition at line 71 of file pvt_conf.h.

12.324.2.20 has_output_file_path

```
std::string Pvt_Conf::has_output_file_path = std::string(".")
```
Definition at line 50 of file pvt_conf.h.

12.324.2.21 kf_use_imu_vel

```
bool Pvt_Conf::kf_use_imu_vel = false
```
Definition at line 131 of file pvt_conf.h.

12.324.2.22 kml_output_enabled

```
bool Pvt_Conf::kml_output_enabled = true
```
Definition at line 102 of file pvt_conf.h.

12.324.2.23 kml_output_path

```
std::string Pvt_Conf::kml_output_path = std::string(".")
```
Definition at line 47 of file pvt_conf.h.

12.324.2.24 kml_rate_ms

```
int32_t Pvt_Conf::kml_rate_ms = 20
```
Definition at line 70 of file pvt_conf.h.

12.324.2.25 log_source_timetag

```
bool Pvt_Conf::log_source_timetag = false
```
Definition at line 112 of file pvt_conf.h.

12.324.2.26 log_source_timetag_file

```
std::string Pvt_Conf::log_source_timetag_file
```
Definition at line 54 of file pvt_conf.h.

12.324.2.27 max_obs_block_rx_clock_offset_ms

```
int32_t Pvt_Conf::max_obs_block_rx_clock_offset_ms = 40
```
Definition at line 77 of file pvt_conf.h.

12.324.2.28 measures_ecef_pos_sd_m

```
double Pvt_Conf::measures_ecef_pos_sd_m = 1.0
```
Definition at line 125 of file pvt_conf.h.

12.324.2.29 measures_ecef_vel_sd_ms

```
double Pvt_Conf::measures_ecef_vel_sd_ms = 0.1
```
Definition at line 126 of file pvt_conf.h.

12.324.2.30 monitor_enabled

```
bool Pvt_Conf::monitor_enabled = false
```
Definition at line 105 of file pvt_conf.h.

12.324.2.31 monitor_ephemeris_enabled

```
bool Pvt_Conf::monitor_ephemeris_enabled = false
```
Definition at line 106 of file pvt_conf.h.

12.324.2.32 nmea_dump_devname

```
std::string Pvt_Conf::nmea_dump_devname
```
Definition at line 39 of file pvt_conf.h.

12.324.2.33 nmea_dump_filename

```
std::string Pvt_Conf::nmea_dump_filename
```
Definition at line 38 of file pvt_conf.h.

12.324.2.34 nmea_output_file_enabled

```
bool Pvt_Conf::nmea_output_file_enabled = true
```
Definition at line 100 of file pvt_conf.h.

12.324.2.35 nmea_output_file_path

```
std::string Pvt_Conf::nmea_output_file_path = std::string(".")
```
Definition at line 46 of file pvt_conf.h.

12.324.2.36 nmea_rate_ms

```
int32_t Pvt_Conf::nmea_rate_ms = 20
```
Definition at line 73 of file pvt_conf.h.

12.324.2.37 ntrip_caster_address

```
std::string Pvt_Conf::ntrip_caster_address
```
Definition at line 55 of file pvt_conf.h.

12.324.2.38 ntrip_client_enabled

```
bool Pvt_Conf::ntrip_client_enabled = false
```
Definition at line 113 of file pvt_conf.h.

12.324.2.39 ntrip_fallback_to_single

```
bool Pvt_Conf::ntrip_fallback_to_single = true
```
Definition at line 114 of file pvt_conf.h.

12.324.2.40 ntrip_gga_period_ms

```
int32_t Pvt_Conf::ntrip_gga_period_ms = 10000
```
Definition at line 84 of file pvt_conf.h.

12.324.2.41 ntrip_max_correction_age_s

```
double Pvt_Conf::ntrip_max_correction_age_s = 5.0
```
Definition at line 91 of file pvt_conf.h.

12.324.2.42 ntrip_mountpoint

```
std::string Pvt_Conf::ntrip_mountpoint
```
Definition at line 56 of file pvt_conf.h.

12.324.2.43 ntrip_password

```
Secure_String Pvt_Conf::ntrip_password
```
Definition at line 62 of file pvt_conf.h.

12.324.2.44 ntrip_password_env

```
Secure_String Pvt_Conf::ntrip_password_env
```
Definition at line 63 of file pvt_conf.h.

12.324.2.45 ntrip_port

```
uint16_t Pvt_Conf::ntrip_port = 2101
```
Definition at line 88 of file pvt_conf.h.

12.324.2.46 ntrip_reconnect_interval_ms

```
int32_t Pvt_Conf::ntrip_reconnect_interval_ms = 10000
```
Definition at line 82 of file pvt_conf.h.

12.324.2.47 ntrip_send_gga

```
bool Pvt_Conf::ntrip_send_gga = true
```
Definition at line 116 of file pvt_conf.h.

12.324.2.48 ntrip_station_id

```
uint16_t Pvt_Conf::ntrip_station_id = 0
```
Definition at line 89 of file pvt_conf.h.

12.324.2.49 ntrip_timeout_ms

```
int32_t Pvt_Conf::ntrip_timeout_ms = 10000
```
Definition at line 81 of file pvt_conf.h.

12.324.2.50 ntrip_tls_enabled

```
bool Pvt_Conf::ntrip_tls_enabled = false
```
Definition at line 115 of file pvt_conf.h.

12.324.2.51 ntrip_username

```
Secure_String Pvt_Conf::ntrip_username
```
Definition at line 61 of file pvt_conf.h.

12.324.2.52 ntrip_version

```
int32_t Pvt_Conf::ntrip_version = 2
```
Definition at line 83 of file pvt_conf.h.

12.324.2.53 observable_interval_ms

```
uint32_t Pvt_Conf::observable_interval_ms = 20
```
Definition at line 66 of file pvt_conf.h.

12.324.2.54 osnma_strict

```
bool Pvt_Conf::osnma_strict = false
```
Definition at line 121 of file pvt_conf.h.

12.324.2.55 output_enabled

```
bool Pvt_Conf::output_enabled = true
```
Definition at line 96 of file pvt_conf.h.

12.324.2.56 output_path

```
std::string Pvt_Conf::output_path = std::string(".")
```
Definition at line 42 of file pvt_conf.h.

12.324.2.57 output_rate_ms

```
int32_t Pvt_Conf::output_rate_ms = 0
```
Definition at line 68 of file pvt_conf.h.

12.324.2.58 protobuf_enabled

```
bool Pvt_Conf::protobuf_enabled = true
```
Definition at line 107 of file pvt_conf.h.

12.324.2.59 ref_gps_week

```
int32_t Pvt_Conf::ref_gps_week = 0
```
Definition at line 78 of file pvt_conf.h.

12.324.2.60 rinex_name

```
std::string Pvt_Conf::rinex_name = std::string("-")
```
Definition at line 36 of file pvt_conf.h.

12.324.2.61 rinex_output_enabled

```
bool Pvt_Conf::rinex_output_enabled = true
```
Definition at line 97 of file pvt_conf.h.

12.324.2.62 rinex_output_path

```
std::string Pvt_Conf::rinex_output_path = std::string(".")
```
Definition at line 43 of file pvt_conf.h.

12.324.2.63 rinex_version

```
int32_t Pvt_Conf::rinex_version = 0
```
Definition at line 74 of file pvt_conf.h.

12.324.2.64 rinexobs_rate_ms

```
int32_t Pvt_Conf::rinexobs_rate_ms = 0
```
Definition at line 75 of file pvt_conf.h.

12.324.2.65 rtcm_dump_devname

```
std::string Pvt_Conf::rtcm_dump_devname
```
Definition at line 40 of file pvt_conf.h.

12.324.2.66 rtcm_msg_rate_ms

```
std::map<int, int> Pvt_Conf::rtcm_msg_rate_ms
```
Definition at line 34 of file pvt_conf.h.

12.324.2.67 rtcm_output_file_enabled

```
bool Pvt_Conf::rtcm_output_file_enabled = true
```
Definition at line 104 of file pvt_conf.h.

12.324.2.68 rtcm_output_file_path

```
std::string Pvt_Conf::rtcm_output_file_path = std::string(".")
```
Definition at line 49 of file pvt_conf.h.

12.324.2.69 rtcm_station_id

```
uint16_t Pvt_Conf::rtcm_station_id = 0
```
Definition at line 87 of file pvt_conf.h.

12.324.2.70 rtcm_tcp_port

```
uint16_t Pvt_Conf::rtcm_tcp_port = 0
```
Definition at line 86 of file pvt_conf.h.

12.324.2.71 rtk_trace_level

```
int Pvt_Conf::rtk_trace_level = 0
```
Definition at line 80 of file pvt_conf.h.

12.324.2.72 show_local_time_zone

```
bool Pvt_Conf::show_local_time_zone = false
```
Definition at line 109 of file pvt_conf.h.

12.324.2.73 signal_enabled_flags

```
uint32_t Pvt_Conf::signal_enabled_flags = 0
```
Definition at line 65 of file pvt_conf.h.

12.324.2.74 system_ecef_pos_sd_m

```
double Pvt_Conf::system_ecef_pos_sd_m = 0.01
```
Definition at line 127 of file pvt_conf.h.

12.324.2.75 system_ecef_vel_sd_ms

```
double Pvt_Conf::system_ecef_vel_sd_ms = 0.001
```
Definition at line 128 of file pvt_conf.h.

12.324.2.76 udp_addresses

```
std::string Pvt_Conf::udp_addresses
```
Definition at line 51 of file pvt_conf.h.

12.324.2.77 udp_eph_addresses

```
std::string Pvt_Conf::udp_eph_addresses
```
Definition at line 53 of file pvt_conf.h.

12.324.2.78 udp_eph_port

```
int Pvt_Conf::udp_eph_port = 0
```
Definition at line 79 of file pvt_conf.h.

12.324.2.79 udp_ports

```
std::string Pvt_Conf::udp_ports
```
Definition at line 52 of file pvt_conf.h.

12.324.2.80 use_e6_for_pvt

```
bool Pvt_Conf::use_e6_for_pvt = true
```
Definition at line 117 of file pvt_conf.h.

12.324.2.81 use_has_corrections

```
bool Pvt_Conf::use_has_corrections = true
```
Definition at line 118 of file pvt_conf.h.

12.324.2.82 use_unhealthy_sats

```
bool Pvt_Conf::use_unhealthy_sats = false
```
Definition at line 119 of file pvt_conf.h.

12.324.2.83 xml_output_enabled

```
bool Pvt_Conf::xml_output_enabled = true
```
Definition at line 103 of file pvt_conf.h.

12.324.2.84 xml_output_path

```
std::string Pvt_Conf::xml_output_path = std::string(".")
```
Definition at line 48 of file pvt_conf.h.
The documentation for this class was generated from the following file:

- pvt_conf.h

12.325 Pvt_Kf Class Reference

Kalman Filter for Position and Velocity.
`#include <pvt_kf.h>`

Public Member Functions

- void **init_Kf** (const arma::vec &p, const arma::vec &v, double update_interval_s, double measures_ecef_↩ pos_sd_m, double measures_ecef_vel_sd_ms, double system_ecef_pos_sd_m, double system_ecef_vel↩ _sd_ms)
- bool **is_initialized** () const
- void **run_Kf** (const arma::vec &p, const arma::vec &v)
- void **get_pv_Kf** (arma::vec &p, arma::vec &v) const
- void **reset_Kf** ()

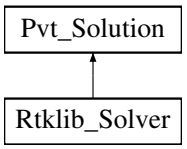
12.325.1 Detailed Description

Kalman Filter for Position and Velocity.
Definition at line 33 of file pvt_kf.h.
The documentation for this class was generated from the following file:

- pvt_kf.h

12.326 Pvt_Solution Class Reference

Base class for a PVT solution.
`#include <pvt_solution.h>`
Inheritance diagram for Pvt_Solution:



Public Member Functions

- virtual double **get_hdop** () const =0
- virtual double **get_vdop** () const =0
- virtual double **get_pdop** () const =0
- virtual double **get_gdop** () const =0
- std::array< double, 3 > **get_rx_pos** () const
- std::array< double, 3 > **get_rx_vel** () const
- boost::posix_time::ptime **get_position_UTC_time** () const
- double get_latitude () const

  Get RX position Latitude WGS84 [deg].
- double get_longitude () const

  Get RX position Longitude WGS84 [deg].
- double get_height () const

  Get RX position height WGS84 [m].
- double get_time_offset_s () const

  Get RX time offset [s].
- double get_clock_drift_ppm () const

  Get the Rx clock drift [ppm].
- double get_speed_over_ground () const

  Get RX speed over ground [m/s].
- double get_course_over_ground () const

  Get RX course over ground [deg].
- int get_num_valid_observations () const

  Get the number of valid pseudorange observations (valid satellites).
- int32_t get_ref_gps_week () const

  Get the reference GPS week used to resolve the mod-1024 week rollover (0 if unset).
- bool **is_valid_position** () const
- void set_rx_pos (const std::array< double, 3 > &pos)

  Set position: X, Y, Z in Cartesian ECEF coordinates [m].
- void set_rx_vel (const std::array< double, 3 > &vel)

  Set velocity: East [m/s], North [m/s], Up [m/s].
- void **set_position_UTC_time** (const boost::posix_time::ptime &pt)
- void set_time_offset_s (double offset)

  Set RX time offset [s].
- void set_clock_drift_ppm (double clock_drift_ppm)

  Set the Rx clock drift [ppm].
- void set_speed_over_ground (double speed_m_s)

  Set RX speed over ground [m/s].
- void set_course_over_ground (double cog_deg)

  Set RX course over ground [deg].
- void **set_valid_position** (bool is_valid)
- void set_num_valid_observations (int num)

  Set the number of valid pseudorange observations (valid satellites).
- void set_ref_gps_week (int32_t ref_week)

  Set the reference GPS week (e.g., derived from the date of signal capture) used to resolve the mod-1024 week rollover in post-processing mode.

12.326.1 Detailed Description

Base class for a PVT solution.
Definition at line 35 of file pvt_solution.h.

12.326.2 Member Function Documentation

12.326.2.1 get_clock_drift_ppm()

```
double Pvt_Solution::get_clock_drift_ppm () const
```

Get the Rx clock drift [ppm].

12.326.2.2 get_course_over_ground()

```
double Pvt_Solution::get_course_over_ground () const
```

Get RX course over ground [deg].

12.326.2.3 get_height()

```
double Pvt_Solution::get_height () const
```

Get RX position height WGS84 [m].

12.326.2.4 get_latitude()

```
double Pvt_Solution::get_latitude () const
```

Get RX position Latitude WGS84 [deg].

12.326.2.5 get_longitude()

```
double Pvt_Solution::get_longitude () const
```

Get RX position Longitude WGS84 [deg].

12.326.2.6 get_num_valid_observations()

```
int Pvt_Solution::get_num_valid_observations () const
```

Get the number of valid pseudorange observations (valid satellites).

12.326.2.7 get_ref_gps_week()

```
int32_t Pvt_Solution::get_ref_gps_week () const
```

Get the reference GPS week used to resolve the mod-1024 week rollover (0 if unset).

12.326.2.8 get_speed_over_ground()

```
double Pvt_Solution::get_speed_over_ground () const
```

Get RX speed over ground [m/s].

12.326.2.9 get_time_offset_s()

```
double Pvt_Solution::get_time_offset_s () const
```

Get RX time offset [s].

12.326.2.10 set_clock_drift_ppm()

```
void Pvt_Solution::set_clock_drift_ppm (
            double clock_drift_ppm)
```

Set the Rx clock drift [ppm].

12.326.2.11 set_course_over_ground()

```
void Pvt_Solution::set_course_over_ground (
            double cog_deg)
```

Set RX course over ground [deg].

12.326.2.12 set_num_valid_observations()

```
void Pvt_Solution::set_num_valid_observations (
            int num)
```

Set the number of valid pseudorange observations (valid satellites).

12.326.2.13 set_ref_gps_week()

```
void Pvt_Solution::set_ref_gps_week (
            int32_t ref_week)
```

Set the reference GPS week (e.g., derived from the date of signal capture) used to resolve the mod-1024 week rollover in post-processing mode.

12.326.2.14 set_rx_pos()

```
void Pvt_Solution::set_rx_pos (
            const std::array< double, 3 > & pos)
```

Set position: X, Y, Z in Cartesian ECEF coordinates [m].

12.326.2.15 set_rx_vel()

```
void Pvt_Solution::set_rx_vel (
            const std::array< double, 3 > & vel)
```

Set velocity: East [m/s], North [m/s], Up [m/s].

12.326.2.16 set_speed_over_ground()

```
void Pvt_Solution::set_speed_over_ground (
            double speed_m_s)
```

Set RX speed over ground [m/s].

12.326.2.17 set_time_offset_s()

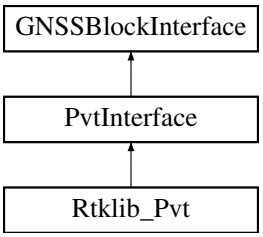
```
void Pvt_Solution::set_time_offset_s (
            double offset)
```

Set RX time offset [s].
The documentation for this class was generated from the following file:

- pvt_solution.h

12.327 PvtInterface Class Reference

This class represents an interface to a PVT block.
`#include <pvt_interface.h>`
Inheritance diagram for PvtInterface:



Public Member Functions

- virtual void **reset** ()=0
- virtual void **clear_ephemeris** ()=0

- virtual std::map< int, Gps_CNAV_Ephemeris > **get_gps_cnav_ephemeris** () const =0
- virtual std::map< int, Glonass_Gnav_Ephemeris > **get_glonass_ephemeris** () const =0
- virtual std::map< int, Glonass_Gnav_Almanac > **get_glonass_almanac** () const =0
- virtual Glonass_Gnav_Utc_Model **get_glonass_utc_model** () const =0
- virtual void **clear_ephemeris_keep_almanac** ()=0
- virtual std::map< int, Gps_Ephemeris > **get_gps_ephemeris** () const =0
- virtual std::map< int, Galileo_Ephemeris > **get_galileo_ephemeris** () const =0
- virtual std::map< int, Beidou_Dnav_Ephemeris > **get_beidou_dnav_ephemeris** () const =0
- virtual std::map< int, Beidou_Cnav1_Ephemeris > get_beidou_cnav1_ephemeris () const
- virtual std::map< int, Beidou_Cnav1_Ephemeris > get_beidou_cnav2_ephemeris () const
- virtual std::map< int, Gps_Almanac > **get_gps_almanac** () const =0
- virtual std::map< int, Galileo_Almanac > **get_galileo_almanac** () const =0
- virtual std::map< int, Beidou_Dnav_Almanac > **get_beidou_dnav_almanac** () const =0
- virtual bool **get_latest_PVT** (double *longitude_deg, double *latitude_deg, double *height_m, double *ground_speed_kmh, double *course_over_ground_deg, time_t *UTC_time)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.327.1 Detailed Description

This class represents an interface to a PVT block.
Abstract class for PVT interfaces, derived from GNSSBlockInterface. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 55 of file pvt_interface.h.

12.327.2 Member Function Documentation

12.327.2.1 get_beidou_cnav1_ephemeris()

```
virtual std::map< int, Beidou_Cnav1_Ephemeris > PvtInterface::get_beidou_cnav1_ephemeris ()
const  [inline], [virtual]
```

Definition at line 74 of file pvt_interface.h.

12.327.2.2 get_beidou_cnav2_ephemeris()

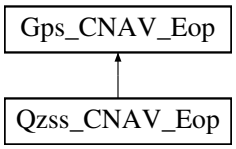
```
virtual std::map< int, Beidou_Cnav1_Ephemeris > PvtInterface::get_beidou_cnav2_ephemeris ()
const  [inline], [virtual]
```

Definition at line 75 of file pvt_interface.h.
The documentation for this class was generated from the following file:

- pvt_interface.h

12.328 Qzss_CNAV_Eop Class Reference

Inheritance diagram for Qzss_CNAV_Eop:



Public Member Functions

- Qzss_CNAV_Eop (const Gps_CNAV_Eop &eop)

Additional Inherited Members

Public Attributes inherited from Gps_CNAV_Eop

- uint32_t PRN {}

  Transmitting satellite PRN.
- double tow {}

  Message transmission time [s of GPS week].
- double t_eop {}

  EOP reference time [s of GPS week].
- double pm_x {}

  X-axis polar motion [arc-seconds].
- double pm_x_dot {}

  X-axis polar motion rate [arc-seconds/day].
- double pm_y {}

  Y-axis polar motion [arc-seconds].
- double pm_y_dot {}

  Y-axis polar motion rate [arc-seconds/day].
- double delta_ut1_gps {}

  UT1-GPS difference [s].
- double delta_ut1_gps_dot {}

  UT1-GPS difference rate [s/day].
- bool valid {}

12.328.1 Detailed Description

Definition at line 30 of file qzss_cnav_eop.h.

12.328.2 Constructor & Destructor Documentation

12.328.2.1 Qzss_CNAV_Eop()

```
Qzss_CNAV_Eop::Qzss_CNAV_Eop (
            const Gps_CNAV_Eop & eop)  [inline], [explicit]
```

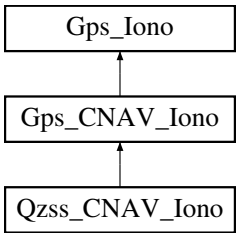
Definition at line 34 of file qzss_cnav_eop.h.

The documentation for this class was generated from the following file:

- qzss_cnav_eop.h

12.329 Qzss_CNAV_Iono Class Reference

This class is a storage for the QZSS CNAV Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
`#include <qzss_cnav_iono.h>`
Inheritance diagram for Qzss_CNAV_Iono:



Public Member Functions

- Qzss_CNAV_Iono ()=default
  Default constructor.
- Qzss_CNAV_Iono (const Gps_CNAV_Iono &iono)

Public Member Functions inherited from Gps_CNAV_Iono

- Gps_CNAV_Iono ()=default
  Default constructor.

Public Member Functions inherited from Gps_Iono

- Gps_Iono ()=default
  Default constructor.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)
  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Additional Inherited Members

Public Attributes inherited from Gps_Iono

- double alpha0 {}
  Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
- double alpha1 {}
  Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].
- double alpha2 {}
  Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].
- double alpha3 {}
  Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].
- double beta0 {}
  Coefficient 0 of a cubic equation representing the period of the model [s].
- double beta1 {}
  Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].
- double beta2 {}
  Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].
- double beta3 {}
  Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].
- bool valid {}
  Valid flag.

12.329.1 Detailed Description

This class is a storage for the QZSS CNAV Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
Message Type 30 broadcasts the Wide Area coefficient set (the Japan area set is broadcast in Message Type 61).
See https://qzss.go.jp/en/technical/ps-is-qzss/
Definition at line 35 of file qzss_cnav_iono.h.

12.329.2 Constructor & Destructor Documentation

12.329.2.1 Qzss_CNAV_Iono() [1/2]

```
Qzss_CNAV_Iono::Qzss_CNAV_Iono ()  [default]
```

Default constructor.
References Gps_CNAV_Iono::Gps_CNAV_Iono().

12.329.2.2 Qzss_CNAV_Iono() [2/2]

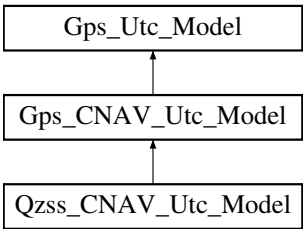
```
Qzss_CNAV_Iono::Qzss_CNAV_Iono (
            const Gps_CNAV_Iono & iono)  [inline], [explicit]
```

Definition at line 39 of file qzss_cnav_iono.h.
The documentation for this class was generated from the following file:

- qzss_cnav_iono.h

12.330 Qzss_CNAV_Utc_Model Class Reference

This class is a storage for the QZSS CNAV UTC model data as described in IS-QZSS-PNT-006.
`#include <qzss_cnav_utc_model.h>`
Inheritance diagram for Qzss_CNAV_Utc_Model:



Public Member Functions

- Qzss_CNAV_Utc_Model ()=default
  Default constructor.
- Qzss_CNAV_Utc_Model (const Gps_CNAV_Utc_Model &utc_model)

Public Member Functions inherited from Gps_CNAV_Utc_Model

- Gps_CNAV_Utc_Model ()=default
  Default constructor.

Public Member Functions inherited from Gps_Utc_Model

- Gps_Utc_Model ()=default
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

Additional Inherited Members

Public Attributes inherited from Gps_Utc_Model

- double A0 {}

  Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s].
- double A1 {}

  1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- double A2 {}

  2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- int32_t tot {}

  Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200M) [s].
- int32_t WN_T {}

  UTC reference week number [weeks].
- int32_t DeltaT_LS {}

  Delta time due to leap seconds [s]. Number of leap seconds since 6-Jan-1980 as transmitted by the GPS almanac.
- int32_t WN_LSF {}

  Week number at the end of which the leap second becomes effective [weeks].
- int32_t DN {}

  Day number (DN) at the end of which the leap second becomes effective [days].
- int32_t DeltaT_LSF {}

  Scheduled future or recent past (relative to NAV message upload) value of the delta time due to leap seconds [s].
- bool valid {}

12.330.1 Detailed Description

This class is a storage for the QZSS CNAV UTC model data as described in IS-QZSS-PNT-006.
The parameters share the GPS CNAV layout, but the broadcast offset refers to UTC(NICT) instead of UTC(USNO).
See https://qzss.go.jp/en/technical/ps-is-qzss/
Definition at line 35 of file qzss_cnav_utc_model.h.

12.330.2 Constructor & Destructor Documentation

12.330.2.1 Qzss_CNAV_Utc_Model() [1/2]

```
Qzss_CNAV_Utc_Model::Qzss_CNAV_Utc_Model ()  [default]
```

Default constructor.
References Gps_CNAV_Utc_Model::Gps_CNAV_Utc_Model().

12.330.2.2 Qzss_CNAV_Utc_Model() [2/2]

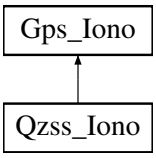
```
Qzss_CNAV_Utc_Model::Qzss_CNAV_Utc_Model (
            const Gps_CNAV_Utc_Model & utc_model)  [inline], [explicit]
```

Definition at line 39 of file qzss_cnav_utc_model.h.
The documentation for this class was generated from the following file:

- qzss_cnav_utc_model.h

12.331 Qzss_Iono Class Reference

This class is a storage for the QZSS Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
`#include <qzss_iono.h>`
Inheritance diagram for Qzss_Iono:



Public Member Functions

- Qzss_Iono ()=default

  Default constructor.
- Qzss_Iono (const Gps_Iono &iono)

Public Member Functions inherited from Gps_Iono

- Gps_Iono ()=default

  Default constructor.
- template<class Archive>
  void serialize (Archive &archive, const unsigned int version)

  Serialize is a boost standard method to be called by the boost XML serialization. Here is used to save the ephemeris data on disk file.

Additional Inherited Members

Public Attributes inherited from Gps_Iono

- double alpha0 {}

  Coefficient 0 of a cubic equation representing the amplitude of the vertical delay [s].
- double alpha1 {}

  Coefficient 1 of a cubic equation representing the amplitude of the vertical delay [s/semi-circle].
- double alpha2 {}

  Coefficient 2 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^2].
- double alpha3 {}

  Coefficient 3 of a cubic equation representing the amplitude of the vertical delay [s(semi-circle)^3].
- double beta0 {}

  Coefficient 0 of a cubic equation representing the period of the model [s].
- double beta1 {}

  Coefficient 1 of a cubic equation representing the period of the model [s/semi-circle].
- double beta2 {}

  Coefficient 2 of a cubic equation representing the period of the model [s(semi-circle)^2].
- double beta3 {}

  Coefficient 3 of a cubic equation representing the period of the model [s(semi-circle)^3].
- bool valid {}

  Valid flag.

12.331.1 Detailed Description

This class is a storage for the QZSS Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.
The coefficients share the GPS LNAV layout, but they are independently determined by QZSS. See https://qzss.go.jp/en/technical/ps-is-qzss/
Definition at line 35 of file qzss_iono.h.

12.331.2 Constructor & Destructor Documentation

12.331.2.1 Qzss_Iono() [1/2]

```
Qzss_Iono::Qzss_Iono ()  [default]
```

Default constructor.
References Gps_Iono::Gps_Iono().

12.331.2.2 Qzss_Iono() [2/2]

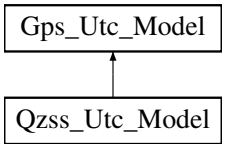
```
Qzss_Iono::Qzss_Iono (
            const Gps_Iono & iono)  [inline], [explicit]
```

Definition at line 39 of file qzss_iono.h.
The documentation for this class was generated from the following file:

- qzss_iono.h

12.332 Qzss_Utc_Model Class Reference

This class is a storage for the QZSS UTC model data as described in IS-QZSS-PNT-006.
`#include <qzss_utc_model.h>`
Inheritance diagram for Qzss_Utc_Model:



Public Member Functions

- Qzss_Utc_Model ()=default

  Default constructor.
- Qzss_Utc_Model (const Gps_Utc_Model &utc_model)

Public Member Functions inherited from Gps_Utc_Model

- Gps_Utc_Model ()=default
- template<class Archive>
  void serialize (Archive &archive, const uint32_t version)

Additional Inherited Members

Public Attributes inherited from Gps_Utc_Model

- double A0 {}

  Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s].
- double A1 {}

  1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- double A2 {}

  2nd order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4 IS-GPS-200M) [s/s]
- int32_t tot {}

  Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4 IS-GPS-200M) [s].
- int32_t WN_T {}

  UTC reference week number [weeks].
- int32_t DeltaT_LS {}

Delta time due to leap seconds [s]. Number of leap seconds since 6-Jan-1980 as transmitted by the GPS almanac.

- int32_t WN_LSF {}

  Week number at the end of which the leap second becomes effective [weeks].

- int32_t DN {}

  Day number (DN) at the end of which the leap second becomes effective [days].

- int32_t DeltaT_LSF {}

  Scheduled future or recent past (relative to NAV message upload) value of the delta time due to leap seconds [s].

- bool valid {}

12.332.1 Detailed Description

This class is a storage for the QZSS UTC model data as described in IS-QZSS-PNT-006.
The parameters share the GPS LNAV layout, but the broadcast offset refers to UTC(NICT) instead of UTC(USNO).
See https://qzss.go.jp/en/technical/ps-is-qzss/
Definition at line 35 of file qzss_utc_model.h.

12.332.2 Constructor & Destructor Documentation

12.332.2.1 Qzss_Utc_Model() [1/2]

```
Qzss_Utc_Model::Qzss_Utc_Model ()  [default]
```

Default constructor.
References Gps_Utc_Model::Gps_Utc_Model().

12.332.2.2 Qzss_Utc_Model() [2/2]

```
Qzss_Utc_Model::Qzss_Utc_Model (
            const Gps_Utc_Model & utc_model)  [inline], [explicit]
```

Definition at line 39 of file qzss_utc_model.h.
The documentation for this class was generated from the following file:

- qzss_utc_model.h

12.333 raw_t Struct Reference

Public Attributes

- gtime_t time
- gtime_t tobs
- obs_t obs
- obs_t obuf
- nav_t nav
- sta_t sta
- int ephsat
- sbsmsg_t sbsmsg
- char msgtype [256]
- unsigned char subfrm [MAXSAT][380]
- lexmsg_t lexmsg
- double lockt [MAXSAT][NFREQ+NEXOBS]
- double icpp [MAXSAT]
- double off [MAXSAT]
- double icpc
- double prCA [MAXSAT]
- double dpCA [MAXSAT]
- unsigned char halfc [MAXSAT][NFREQ+NEXOBS]
- char freqn [MAXOBS]
- int nbyte

- int len
- int iod
- int tod
- int tbase
- int flag
- int outtype
- unsigned char buff [MAXRAWLEN]
- char opt [256]
- double receive_time
- unsigned int plen
- unsigned int pbyte
- unsigned int page
- unsigned int reply
- int week
- unsigned char pbuff [255+4+2]

12.333.1 Detailed Description

Definition at line 1281 of file rtklib.h.

12.333.2 Member Data Documentation

12.333.2.1 buff

```
unsigned char raw_t::buff[MAXRAWLEN]
```

Definition at line 1306 of file rtklib.h.

12.333.2.2 dpCA

```
double raw_t::dpCA[MAXSAT]
```

Definition at line 1296 of file rtklib.h.

12.333.2.3 ephsat

```
int raw_t::ephsat
```

Definition at line 1289 of file rtklib.h.

12.333.2.4 flag

```
int raw_t::flag
```

Definition at line 1304 of file rtklib.h.

12.333.2.5 freqn

```
char raw_t::freqn[MAXOBS]
```

Definition at line 1298 of file rtklib.h.

12.333.2.6 halfc

```
unsigned char raw_t::halfc[MAXSAT][NFREQ+NEXOBS]
```

Definition at line 1297 of file rtklib.h.

12.333.2.7 icpc

```
double raw_t::icpc
```

Definition at line 1295 of file rtklib.h.

12.333.2.8 icpp

```
double raw_t::icpp[MAXSAT]
```
Definition at line 1295 of file rtklib.h.

12.333.2.9 iod

```
int raw_t::iod
```
Definition at line 1301 of file rtklib.h.

12.333.2.10 len

```
int raw_t::len
```
Definition at line 1300 of file rtklib.h.

12.333.2.11 lexmsg

```
lexmsg_t raw_t::lexmsg
```
Definition at line 1293 of file rtklib.h.

12.333.2.12 lockt

```
double raw_t::lockt[MAXSAT][NFREQ+NEXOBS]
```
Definition at line 1294 of file rtklib.h.

12.333.2.13 msgtype

```
char raw_t::msgtype[256]
```
Definition at line 1291 of file rtklib.h.

12.333.2.14 nav

```
nav_t raw_t::nav
```
Definition at line 1287 of file rtklib.h.

12.333.2.15 nbyte

```
int raw_t::nbyte
```
Definition at line 1299 of file rtklib.h.

12.333.2.16 obs

```
obs_t raw_t::obs
```
Definition at line 1285 of file rtklib.h.

12.333.2.17 obuf

```
obs_t raw_t::obuf
```
Definition at line 1286 of file rtklib.h.

12.333.2.18 off

```
double raw_t::off[MAXSAT]
```
Definition at line 1295 of file rtklib.h.

12.333.2.19 opt

```
char raw_t::opt[256]
```
Definition at line 1307 of file rtklib.h.

12.333.2.20 outtype

```
int raw_t::outtype
```
Definition at line 1305 of file rtklib.h.

12.333.2.21 page

```
unsigned int raw_t::page
```
Definition at line 1311 of file rtklib.h.

12.333.2.22 pbuff

```
unsigned char raw_t::pbuff[255+4+2]
```
Definition at line 1314 of file rtklib.h.

12.333.2.23 pbyte

```
unsigned int raw_t::pbyte
```
Definition at line 1310 of file rtklib.h.

12.333.2.24 plen

```
unsigned int raw_t::plen
```
Definition at line 1309 of file rtklib.h.

12.333.2.25 prCA

```
double raw_t::prCA[MAXSAT]
```
Definition at line 1296 of file rtklib.h.

12.333.2.26 receive_time

```
double raw_t::receive_time
```
Definition at line 1308 of file rtklib.h.

12.333.2.27 reply

```
unsigned int raw_t::reply
```
Definition at line 1312 of file rtklib.h.

12.333.2.28 sbsmsg

```
sbsmsg_t raw_t::sbsmsg
```
Definition at line 1290 of file rtklib.h.

12.333.2.29 sta

```
sta_t raw_t::sta
```
Definition at line 1288 of file rtklib.h.

12.333.2.30 subfrm

```
unsigned char raw_t::subfrm[MAXSAT][380]
```
Definition at line 1292 of file rtklib.h.

12.333.2.31 tbase

```
int raw_t::tbase
```
Definition at line 1303 of file rtklib.h.

12.333.2.32 time

gtime_t raw_t::time
Definition at line 1283 of file rtklib.h.

12.333.2.33 tobs

gtime_t raw_t::tobs
Definition at line 1284 of file rtklib.h.

12.333.2.34 tod

int raw_t::tod
Definition at line 1302 of file rtklib.h.

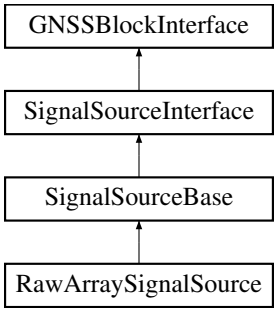
12.333.2.35 week

int raw_t::week
Definition at line 1313 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.334 RawArraySignalSource Class Reference

This class reads samples from an antenna array RF front-end signal sampler.
#include <raw_array_signal_source.h>
Inheritance diagram for RawArraySignalSource:



Public Member Functions

- **RawArraySignalSource** (const ConfigurationInterface *configuration, std::string role, unsigned int in_↩ stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.334.1 Detailed Description

This class reads samples from an antenna array RF front-end signal sampler.
Definition at line 42 of file raw_array_signal_source.h.

12.334.2 Member Function Documentation

12.334.2.1 connect()

```
void RawArraySignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.334.2.2 disconnect()

```
void RawArraySignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.334.2.3 get_left_block()

```
gr::basic_block_sptr RawArraySignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.334.2.4 get_right_block()

```
gr::basic_block_sptr RawArraySignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.334.2.5 item_size()

```
size_t RawArraySignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 51 of file raw_array_signal_source.h.
The documentation for this class was generated from the following file:

- raw_array_signal_source.h

12.335 ReedSolomon Class Reference

Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where k=255-nroots is the information vector length and d=nroots+1 is the minimum Hamming distance, with symbols of 8 bits. It allows shortened RS codes.
`#include <reed_solomon.h>`

Public Member Functions

- ReedSolomon (const std::string &gnss_signal=std::string())

  Default constructor. Constructs a Reed Solomon object. The encode_with_generator_poly and encode_with_↩ generator_matrix methods are available for testing purposes.
- ReedSolomon (int nroots, int minpoly, int prim, int fcr, int pad=0, int shortening=0, const std::vector< uint8_t > &genpoly_coeff=std::vector< uint8_t >{}, const std::vector< std::vector< uint8_t > > &gen_matrix=std::↩ vector< std::vector< uint8_t > >{})

  Custom constructor for RS(255, 255-nroots, nroots+1). Parameters:
- int decode (std::vector< uint8_t > &data_to_decode, const std::vector< int > &erasure_positions=std::↩ vector< int >{}) const

  Decode an encoded block.
- std::vector< uint8_t > encode_with_generator_matrix (const std::vector< uint8_t > &data_to_encode) const

  Encode data with the generator matrix (for testing purposes).
- std::vector< uint8_t > encode_with_generator_poly (const std::vector< uint8_t > &data_to_encode) const

  Encode data with the generator polynomial (for testing purposes).

12.335.1 Detailed Description

Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where k=255-nroots is the information vector length and d=nroots+1 is the minimum Hamming distance, with symbols of 8 bits. It allows shortened RS codes.
Definition at line 40 of file reed_solomon.h.

12.335.2 Constructor & Destructor Documentation

12.335.2.1 ReedSolomon() [1/2]

```
ReedSolomon::ReedSolomon (
            const std::string & gnss_signal = std::string())  [explicit]
```

Default constructor. Constructs a Reed Solomon object. The encode_with_generator_poly and encode_with_↩ generator_matrix methods are available for testing purposes.
gnss_signal: empty or "E6B" sets the Galileo E6B RS parameters. "E1B" sets the Galileo E1B (INAV) RS parameters.

12.335.2.2 ReedSolomon() [2/2]

```
ReedSolomon::ReedSolomon (
            int nroots,
            int minpoly,
            int prim,
            int fcr,
            int pad = 0,
            int shortening = 0,
            const std::vector< uint8_t > & genpoly_coeff = std::vector< uint8_t >{},
            const std::vector< std::vector< uint8_t > > & gen_matrix = std::vector< std::vector< uint8_t >
```

Custom constructor for RS(255, 255-nroots, nroots+1). Parameters:
nroots - the number of roots in the RS code generator polynomial, which is the same as the number of parity symbols in a block.
minpoly - primitive polynomial.
prim - the primitive root of the generator polynomial.

fcr - first consecutive root of the Reed-Solomon generator polynomial.
pad - the number of pad symbols in a block. If not defined, it defaults to 0.
shortening - value of the shortening parameter. Defaults to 0. If different to 0, it defines a shortened RS encoder/↩ decoder.
genpoly_coeff - a vector of (nroots+1) elements containing the generator polynomial coefficients. Only used for encoding. Defaults to empty. If defined, the encode_with_generator_poly method can be used.
gen_matrix - a (255-shortening)x(255-nroots-shortening) matrix containing the elements of the generator matrix. Only used for encoding. Defaults to empty. If defined, the encode_with_generator_matrix method can be used.

12.335.3 Member Function Documentation

12.335.3.1 decode()

```
int ReedSolomon::decode (
            std::vector< uint8_t > & data_to_decode,
            const std::vector< int > & erasure_positions = std::vector< int >{}) const
```

Decode an encoded block.
The decoded symbols are at the first 255-nroots-shortening elements of the data_to_decode vector.
The second parameter is optional, and contains a vector of erasure positions to be passed to the decoding algorithm. Defaults to empty.
Returns the number of corrected errors, or -1 if decoding failed.

12.335.3.2 encode_with_generator_matrix()

```
std::vector< uint8_t > ReedSolomon::encode_with_generator_matrix (
            const std::vector< uint8_t > & data_to_encode) const
```

Encode data with the generator matrix (for testing purposes).
Returns the encoded vector. It is set to all zeros if the generator matrix is not defined.

12.335.3.3 encode_with_generator_poly()

```
std::vector< uint8_t > ReedSolomon::encode_with_generator_poly (
            const std::vector< uint8_t > & data_to_encode) const
```

Encode data with the generator polynomial (for testing purposes).
Returns the encoded vector. It is set to all zeros if the generator polynomial is not defined.
The documentation for this class was generated from the following file:

- reed_solomon.h

12.336 Rinex_Printer Class Reference

Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.
`#include <rinex_printer.h>`

Public Member Functions

- Rinex_Printer (uint32_t signal_enabled_flags, int version=3, const std::string &base_path=".", const std::↩ string &base_name="-", int32_t ref_gps_week=0)

  Constructor. Creates GNSS Navigation and Observables RINEX files.
- ∼Rinex_Printer () noexcept

  Destructor. Removes created files if empty.
- void print_rinex_annotation (const Rtklib_Solver ∗pvt_solver, const std::map< int, Gnss_Synchro > &gnss↩ _observables_map, double rx_time, bool flag_write_RINEX_obs_output)

  Print RINEX annotation. If it is the first annotation, it also prints the RINEX headers for navigation and observation files. If it is not the first annotation, it only annotates the observation, and updates the navigation header if UTC data was not available when writing it for the first time.
- void log_rinex_nav_gps_nav (const std::map< int32_t, Gps_Ephemeris > &new_eph)

  Print RINEX annotation for GPS NAV message.

- void log_rinex_nav_gps_cnav (const std::map< int32_t, Gps_CNAV_Ephemeris > &new_cnav_eph)

  Print RINEX 4 annotation for GPS/QZSS CNAV message.
- void log_rinex_nav_gal_nav (const std::map< int32_t, Galileo_Ephemeris > &new_gal_eph)

  Print RINEX annotation for Galileo NAV message.
- void log_rinex_nav_glo_gnav (const std::map< int32_t, Glonass_Gnav_Ephemeris > &new_glo_eph)

  Print RINEX annotation for Glonass GNAV message.
- void log_rinex_nav_bds_dnav (const std::map< int32_t, Beidou_Dnav_Ephemeris > &new_bds_eph)

  Print RINEX annotation for BeiDou DNAV message.
- void log_rinex_nav_bds_cnav1 (const std::map< int32_t, Beidou_Cnav1_Ephemeris > &new_bds_eph, const std::map< int32_t, Bds3_B1c_PageData > &bds_page_data)

  Print RINEX navigation records for BeiDou B-CNAV1 ephemerides.
- void log_rinex_nav_bds_cnav2 (const std::map< int32_t, Beidou_Cnav1_Ephemeris > &new_bds_eph)

  Print RINEX navigation records for BeiDou B-CNAV2 ephemerides.
- bool is_rinex_header_written () const

  Returns true is the RINEX file headers are already written.
- std::vector< std::string > get_navfilename () const

  Returns name of RINEX navigation file(s).
- std::string get_obsfilename () const

  Returns name of RINEX observation file.

12.336.1 Detailed Description

Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.
Definition at line 84 of file rinex_printer.h.

12.336.2 Constructor & Destructor Documentation

12.336.2.1 Rinex_Printer()

```
Rinex_Printer::Rinex_Printer (
            uint32_t signal_enabled_flags,
            int version = 3,
            const std::string & base_path = ".",
            const std::string & base_name = "-",
            int32_t ref_gps_week = 0)  [explicit]
```

Constructor. Creates GNSS Navigation and Observables RINEX files.

12.336.2.2 ~Rinex_Printer()

```
Rinex_Printer::~Rinex_Printer ()  [noexcept]
```

Destructor. Removes created files if empty.

12.336.3 Member Function Documentation

12.336.3.1 get_navfilename()

```
std::vector< std::string > Rinex_Printer::get_navfilename () const  [inline]
```

Returns name of RINEX navigation file(s).
Definition at line 171 of file rinex_printer.h.

12.336.3.2 get_obsfilename()

```
std::string Rinex_Printer::get_obsfilename () const  [inline]
```

Returns name of RINEX observation file.
Definition at line 179 of file rinex_printer.h.

12.336.3.3 is_rinex_header_written()

```
bool Rinex_Printer::is_rinex_header_written () const  [inline]
```

Returns true is the RINEX file headers are already written.
Definition at line 163 of file rinex_printer.h.
References is_rinex_header_written().
Referenced by is_rinex_header_written().

12.336.3.4 log_rinex_nav_bds_cnav1()

```
void Rinex_Printer::log_rinex_nav_bds_cnav1 (
            const std::map< int32_t, Beidou_Cnav1_Ephemeris > & new_bds_eph,
            const std::map< int32_t, Bds3_B1c_PageData > & bds_page_data)
```

Print RINEX navigation records for BeiDou B-CNAV1 ephemerides.
RINEX 4 writes proper CNV1 records (Table A24), taking SISAI/SISMAI/ integrity and t_op from the last decoded SF3 page when it carries them. RINEX 3 has no CNAV1 representation, so records are written as D1-style stand-ins: sqrt(A0) in the sqrtA field with the Adot term dropped, IODE/IODC in AODE/AODC, and TGD_B1Cp/TGD_B2ap mapped into TGD1/TGD2.
References log_rinex_nav_bds_cnav1().
Referenced by log_rinex_nav_bds_cnav1().

12.336.3.5 log_rinex_nav_bds_cnav2()

```
void Rinex_Printer::log_rinex_nav_bds_cnav2 (
            const std::map< int32_t, Beidou_Cnav1_Ephemeris > & new_bds_eph)
```

Print RINEX navigation records for BeiDou B-CNAV2 ephemerides.
RINEX 4 writes CNV2 records (Table A25) with ISC_B2ad in the second field of orbit line 7. RINEX 3 has no CNV2 representation; records are omitted rather than written as CNV1 or D1 stand-ins.
References log_rinex_nav_bds_cnav2().
Referenced by log_rinex_nav_bds_cnav2().

12.336.3.6 log_rinex_nav_bds_dnav()

```
void Rinex_Printer::log_rinex_nav_bds_dnav (
            const std::map< int32_t, Beidou_Dnav_Ephemeris > & new_bds_eph)
```

Print RINEX annotation for BeiDou DNAV message.
References log_rinex_nav_bds_dnav().
Referenced by log_rinex_nav_bds_dnav().

12.336.3.7 log_rinex_nav_gal_nav()

```
void Rinex_Printer::log_rinex_nav_gal_nav (
            const std::map< int32_t, Galileo_Ephemeris > & new_gal_eph)
```

Print RINEX annotation for Galileo NAV message.
References log_rinex_nav_gal_nav().
Referenced by log_rinex_nav_gal_nav().

12.336.3.8 log_rinex_nav_glo_gnav()

```
void Rinex_Printer::log_rinex_nav_glo_gnav (
            const std::map< int32_t, Glonass_Gnav_Ephemeris > & new_glo_eph)
```

Print RINEX annotation for Glonass GNAV message.
References log_rinex_nav_glo_gnav().
Referenced by log_rinex_nav_glo_gnav().

12.336.3.9 log_rinex_nav_gps_cnav()

```
void Rinex_Printer::log_rinex_nav_gps_cnav (
            const std::map< int32_t, Gps_CNAV_Ephemeris > & new_cnav_eph)
```

Print RINEX 4 annotation for GPS/QZSS CNAV message.

References log_rinex_nav_gps_cnav().
Referenced by log_rinex_nav_gps_cnav().

12.336.3.10 log_rinex_nav_gps_nav()

```
void Rinex_Printer::log_rinex_nav_gps_nav (
            const std::map< int32_t, Gps_Ephemeris > & new_eph)
```

Print RINEX annotation for GPS NAV message.
References log_rinex_nav_gps_nav().
Referenced by log_rinex_nav_gps_nav().

12.336.3.11 print_rinex_annotation()

```
void Rinex_Printer::print_rinex_annotation (
            const Rtklib_Solver * pvt_solver,
            const std::map< int, Gnss_Synchro > & gnss_observables_map,
            double rx_time,
            bool flag_write_RINEX_obs_output)
```

Print RINEX annotation. If it is the first annotation, it also prints the RINEX headers for navigation and observation files. If it is not the first annotation, it only annotates the observation, and updates the navigation header if UTC data was not available when writing it for the first time.
References print_rinex_annotation().
Referenced by print_rinex_annotation().
The documentation for this class was generated from the following file:

- rinex_printer.h

12.337 Rtcm Class Reference

This class implements the generation and reading of some Message Types defined in the RTCM 3.2 Standard, plus some utilities to handle messages.
`#include <rtcm.h>`

Public Member Functions

- Rtcm (uint16_t port=2101)

  Default constructor that sets TCP port of the RTCM message server and RTCM Station ID. 2101 is the standard RTCM port according to the Internet Assigned Numbers Authority (IANA). See https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml.
- std::string print_MT1001 (const Gps_Ephemeris &gps_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints message type 1001 (L1-Only GPS RTK Observables).
- std::string print_MT1002 (const Gps_Ephemeris &gps_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints message type 1002 (Extended L1-Only GPS RTK Observables).
- std::string print_MT1003 (const Gps_Ephemeris &ephL1, const Gps_CNAV_Ephemeris &ephL2, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints message type 1003 (L1 & L2 GPS RTK Observables).
- std::string print_MT1004 (const Gps_Ephemeris &ephL1, const Gps_CNAV_Ephemeris &ephL2, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints message type 1004 (Extended L1 & L2 GPS RTK Observables).
- std::string print_MT1005 (uint32_t ref_id, double ecef_x, double ecef_y, double ecef_z, bool gps, bool glonass, bool galileo, bool non_physical, bool single_oscillator, uint32_t quarter_cycle_indicator)

  Prints message type 1005 (Stationary Antenna Reference Point).
- int32_t read_MT1005 (const std::string &message, uint32_t &ref_id, double &ecef_x, double &ecef_y, double &ecef_z, bool &gps, bool &glonass, bool &galileo)

Verifies and reads messages of type 1005 (Stationary Antenna Reference Point). Returns 1 if anything goes wrong, 0 otherwise.

- std::string print_MT1006 (uint32_t ref_id, double ecef_x, double ecef_y, double ecef_z, bool gps, bool glonass, bool galileo, bool non_physical, bool single_oscillator, uint32_t quarter_cycle_indicator, double height)

  Prints message type 1006 (Stationary Antenna Reference Point, with Height Information).
- std::string print_MT1005_test ()

  For testing purposes.
- std::string print_MT1008 (uint32_t ref_id, const std::string &antenna_descriptor, uint32_t antenna_setup_id, const std::string &antenna_serial_number)

  Prints message type 1008 (Antenna Descriptor & Serial Number).
- std::string print_MT1009 (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints L1-Only GLONASS RTK Observables.
- std::string print_MT1010 (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints Extended L1-Only GLONASS RTK Observables.
- std::string print_MT1011 (const Glonass_Gnav_Ephemeris &glonass_gnav_ephL1, const Glonass_Gnav_Ephemeris &glonass_gnav_ephL2, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints L1&L2 GLONASS RTK Observables.
- std::string print_MT1012 (const Glonass_Gnav_Ephemeris &glonass_gnav_ephL1, const Glonass_Gnav_Ephemeris &glonass_gnav_ephL2, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint16_t station_id)

  Prints Extended L1&L2 GLONASS RTK Observables.
- std::string print_MT1019 (const Gps_Ephemeris &gps_eph)

  Prints message type 1019 (GPS Ephemeris), should be broadcast in the event that the IODC does not match the IODE, and every 2 minutes.
- int32_t read_MT1019 (const std::string &message, Gps_Ephemeris &gps_eph) const

  Verifies and reads messages of type 1019 (GPS Ephemeris). Returns 1 if anything goes wrong, 0 otherwise.
- std::string print_MT1020 (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, const Glonass_Gnav_Utc_Model &glonass_gnav_utc_model)

  Prints message type 1020 (GLONASS Ephemeris).
- int32_t read_MT1020 (const std::string &message, Glonass_Gnav_Ephemeris &glonass_gnav_eph, Glonass_Gnav_Utc_Model &glonass_gnav_utc_model) const

  Verifies and reads messages of type 1020 (GLONASS Ephemeris).
- std::string print_MT1029 (uint32_t ref_id, const Gps_Ephemeris &gps_eph, double obs_time, const std::string &message)

  Prints message type 1029 (Unicode Text String).
- std::string print_MT1045 (const Galileo_Ephemeris &gal_eph)

  Prints message type 1045 (Galileo Ephemeris), should be broadcast every 2 minutes.
- int32_t read_MT1045 (const std::string &message, Galileo_Ephemeris &gal_eph) const

  Verifies and reads messages of type 1045 (Galileo Ephemeris). Returns 1 if anything goes wrong, 0 otherwise.
- std::string print_MSM_1 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM1 (Compact GNSS observables).
- std::string print_MSM_2 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM2 (Compact GNSS phaseranges).

- std::string print_MSM_3 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM3 (Compact GNSS pseudoranges and phaseranges).
- std::string print_MSM_4 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM4 (Full GNSS pseudoranges and phaseranges plus CNR).
- std::string print_MSM_5 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM5 (Full GNSS pseudoranges, phaseranges, phaserange rate and CNR).
- std::string print_MSM_6 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM6 (Full GNSS pseudoranges and phaseranges plus CNR, high resolution).
- std::string print_MSM_7 (const Gps_Ephemeris &gps_eph, const Gps_CNAV_Ephemeris &gps_cnav_eph, const Galileo_Ephemeris &gal_eph, const Glonass_Gnav_Ephemeris &glo_gnav_eph, double obs_time, const std::map< int32_t, Gnss_Synchro > &observables, uint32_t ref_id, uint32_t clock_steering_indicator, uint32_t external_clock_indicator, int32_t smooth_int, bool divergence_free, bool more_messages)

  Prints messages of type MSM7 (Full GNSS pseudoranges, phaseranges, phaserange rate and CNR, high resolution).
- std::string print_MT1057 (const Galileo_HAS_data &has_data, bool ssr_multiple_msg_indicator=false)

  Prints message type 1057 (SSR GPS Orbit Correction).
- std::string print_MT1058 (const Galileo_HAS_data &has_data, bool use_clock_subset=false, bool ssr_↩ multiple_msg_indicator=false)

  Prints message type 1058 (SSR GPS Clock Correction).
- std::string print_MT1059 (const Galileo_HAS_data &has_data, bool ssr_multiple_msg_indicator=false)

  Prints message type 1059 (SSR GPS Satellite Code Bias).
- std::string print_MT1060 (const Galileo_HAS_data &has_data, bool ssr_multiple_msg_indicator=false)

  Prints message type 1060 (SSR GPS Combined Orbit and Clock Correction).
- std::vector< std::string > print_IGM01 (const Galileo_HAS_data &has_data)

  Prints messages of type IGM01 (SSR Orbit Correction).
- std::vector< std::string > print_IGM02 (const Galileo_HAS_data &has_data, bool use_clock_subset=false)

  Prints messages of type IGM02 (SSR Clock Correction).
- std::vector< std::string > print_IGM03 (const Galileo_HAS_data &has_data)

  Prints messages of type IGM03 (SSR Combined Orbit and Clock Correction).
- std::vector< std::string > print_IGM05 (const Galileo_HAS_data &has_data)

  Prints messages of type IGM05 (SSR Bias Correction).
- uint32_t lock_time (const Gps_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)

  Returns the time period in which GPS L1 signals have been continually tracked.
- uint32_t lock_time (const Gps_CNAV_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_↩ synchro)

  Returns the time period in which GPS L2 signals have been continually tracked.
- uint32_t lock_time (const Galileo_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)

  Returns the time period in which Galileo signals have been continually tracked.
- uint32_t lock_time (const Glonass_Gnav_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss↩ _synchro)

  Locks time period in which GLONASS signals have been continually tracked.
- std::string bin_to_hex (const std::string &s) const

Returns a string of hexadecimal symbols from a string of binary symbols.

- std::string hex_to_bin (const std::string &s) const

  Returns a string of binary symbols from a string of hexadecimal symbols.
- std::string bin_to_binary_data (const std::string &s) const

  Returns a string of binary data from a string of binary symbols.
- std::string binary_data_to_bin (const std::string &s) const

  Returns a string of binary symbols from a string of binary data.
- uint32_t bin_to_uint (const std::string &s) const

  Returns an uint32_t from a string of binary symbols.
- int32_t **bin_to_int** (const std::string &s) const
- double bin_to_double (const std::string &s) const

  Returns double from a string of binary symbols.
- int32_t **bin_to_sint** (const std::string &s) const
- uint64_t hex_to_uint (const std::string &s) const

  Returns an uint64_t from a string of hexadecimal symbols.
- int64_t hex_to_int (const std::string &s) const

  Returns a int64_t from a string of hexadecimal symbols.
- bool check_CRC (const std::string &message) const

  Checks that the CRC of a RTCM package is correct.
- void run_server ()

  Starts running the server.
- void stop_server ()

  Stops the server.
- void send_message (const std::string &msg)

  Sends a message through the server to all connected clients.
- bool is_server_running () const

  Returns true if the server is running, false otherwise.

12.337.1 Detailed Description

This class implements the generation and reading of some Message Types defined in the RTCM 3.2 Standard, plus some utilities to handle messages.
Generation of the following Message Types: 1001, 1002, 1003, 1004, 1005, 1006, 1008, 1019, 1020, 1029, 1045, 1057, 1058, 1059, 1060
Decoding of the following Message Types: 1019, 1045
Generation of the following Multiple Signal Messages: MSM1 (message types 1071, 1091) MSM2 (message types 1072, 1092) MSM3 (message types 1073, 1093) MSM4 (message types 1074, 1094) MSM5 (message types 1075, 1095) MSM6 (message types 1076, 1096) MSM7 (message types 1077, 1097)
RTCM 3 message format (size in bits): +--------—+-------—+----------—+------------------—+--------—+ | preamble | 000000 | length | data message | parity | +----------—+-------—+----------—+------------------—+--------—+ |<– 8 –->|<- 6 -->|<– 10 –->|<— length x 8 –->|<– 24 –->| +--------—+-------—+----------—+------------------—+--------—+
(C) Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es
Definition at line 98 of file rtcm.h.

12.337.2 Constructor & Destructor Documentation

12.337.2.1 Rtcm()

```
Rtcm::Rtcm (
            uint16_t port = 2101)  [explicit]
```

Default constructor that sets TCP port of the RTCM message server and RTCM Station ID. 2101 is the standard RTCM port according to the Internet Assigned Numbers Authority (IANA). See https://www.iana.org/↩assignments/service-names-port-numbers/service-names-port-numbers.xml.
References Rtcm().
Referenced by Rtcm().

12.337.3 Member Function Documentation

12.337.3.1 bin_to_binary_data()

```
std::string Rtcm::bin_to_binary_data (
            const std::string & s) const
```

Returns a string of binary data from a string of binary symbols.
References bin_to_binary_data().
Referenced by bin_to_binary_data().

12.337.3.2 bin_to_double()

```
double Rtcm::bin_to_double (
            const std::string & s) const
```

Returns double from a string of binary symbols.
References bin_to_double().
Referenced by bin_to_double().

12.337.3.3 bin_to_hex()

```
std::string Rtcm::bin_to_hex (
            const std::string & s) const
```

Returns a string of hexadecimal symbols from a string of binary symbols.
References bin_to_hex().
Referenced by bin_to_hex().

12.337.3.4 bin_to_uint()

```
uint32_t Rtcm::bin_to_uint (
            const std::string & s) const
```

Returns an uint32_t from a string of binary symbols.
References bin_to_uint().
Referenced by bin_to_uint().

12.337.3.5 binary_data_to_bin()

```
std::string Rtcm::binary_data_to_bin (
            const std::string & s) const
```

Returns a string of binary symbols from a string of binary data.
References binary_data_to_bin().
Referenced by binary_data_to_bin().

12.337.3.6 check_CRC()

```
bool Rtcm::check_CRC (
            const std::string & message) const
```

Checks that the CRC of a RTCM package is correct.
References check_CRC().
Referenced by check_CRC().

12.337.3.7 hex_to_bin()

```
std::string Rtcm::hex_to_bin (
            const std::string & s) const
```

Returns a string of binary symbols from a string of hexadecimal symbols.
References hex_to_bin().
Referenced by hex_to_bin().

12.337.3.8 hex_to_int()

```
int64_t Rtcm::hex_to_int (
            const std::string & s) const
```

Returns a int64_t from a string of hexadecimal symbols.
References hex_to_int().
Referenced by hex_to_int().

12.337.3.9 hex_to_uint()

```
uint64_t Rtcm::hex_to_uint (
            const std::string & s) const
```

Returns an uint64_t from a string of hexadecimal symbols.
References hex_to_uint().
Referenced by hex_to_uint().

12.337.3.10 is_server_running()

```
bool Rtcm::is_server_running () const
```

Returns true if the server is running, false otherwise.
References is_server_running().
Referenced by is_server_running().

12.337.3.11 lock_time() [1/4]

```
uint32_t Rtcm::lock_time (
            const Galileo_Ephemeris & eph,
            double obs_time,
            const Gnss_Synchro & gnss_synchro)
```

Returns the time period in which Galileo signals have been continually tracked.
References lock_time().

12.337.3.12 lock_time() [2/4]

```
uint32_t Rtcm::lock_time (
            const Glonass_Gnav_Ephemeris & eph,
            double obs_time,
            const Gnss_Synchro & gnss_synchro)
```

Locks time period in which GLONASS signals have been continually tracked.

Note

Code added as part of GSoC 2017 program

Parameters

eph	GLONASS GNAV Broadcast Ephemeris
obs_time	Time of observation at the moment of printing
observables	Set of observables as defined by the platform

Returns

Returns the time period in which GLONASS signals have been continually tracked.

References lock_time().

12.337.3.13 lock_time() [3/4]

```
uint32_t Rtcm::lock_time (
            const Gps_CNAV_Ephemeris & eph,
            double obs_time,
            const Gnss_Synchro & gnss_synchro)
```

Returns the time period in which GPS L2 signals have been continually tracked.
References lock_time().

12.337.3.14 lock_time() [4/4]

```
uint32_t Rtcm::lock_time (
            const Gps_Ephemeris & eph,
            double obs_time,
            const Gnss_Synchro & gnss_synchro)
```

Returns the time period in which GPS L1 signals have been continually tracked.
References lock_time().
Referenced by lock_time(), lock_time(), lock_time(), and lock_time().

12.337.3.15 print_IGM01()

```
std::vector< std::string > Rtcm::print_IGM01 (
            const Galileo_HAS_data & has_data)
```

Prints messages of type IGM01 (SSR Orbit Correction).
References print_IGM01().
Referenced by print_IGM01().

12.337.3.16 print_IGM02()

```
std::vector< std::string > Rtcm::print_IGM02 (
            const Galileo_HAS_data & has_data,
            bool use_clock_subset = false)
```

Prints messages of type IGM02 (SSR Clock Correction).
References print_IGM02().
Referenced by print_IGM02().

12.337.3.17 print_IGM03()

```
std::vector< std::string > Rtcm::print_IGM03 (
            const Galileo_HAS_data & has_data)
```

Prints messages of type IGM03 (SSR Combined Orbit and Clock Correction).
References print_IGM03().
Referenced by print_IGM03().

12.337.3.18 print_IGM05()

```
std::vector< std::string > Rtcm::print_IGM05 (
            const Galileo_HAS_data & has_data)
```

Prints messages of type IGM05 (SSR Bias Correction).
References print_IGM05().
Referenced by print_IGM05().

12.337.3.19 print_MSM_1()

```
std::string Rtcm::print_MSM_1 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
```

```
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM1 (Compact GNSS observables).
References print_MSM_1().
Referenced by print_MSM_1().

12.337.3.20 print_MSM_2()

```
std::string Rtcm::print_MSM_2 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM2 (Compact GNSS phaseranges).
References print_MSM_2().
Referenced by print_MSM_2().

12.337.3.21 print_MSM_3()

```
std::string Rtcm::print_MSM_3 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM3 (Compact GNSS pseudoranges and phaseranges).
References print_MSM_3().
Referenced by print_MSM_3().

12.337.3.22 print_MSM_4()

```
std::string Rtcm::print_MSM_4 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
```

```
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM4 (Full GNSS pseudoranges and phaseranges plus CNR).
References print_MSM_4().
Referenced by print_MSM_4().

12.337.3.23 print_MSM_5()

```
std::string Rtcm::print_MSM_5 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM5 (Full GNSS pseudoranges, phaseranges, phaserange rate and CNR).
References print_MSM_5().
Referenced by print_MSM_5().

12.337.3.24 print_MSM_6()

```
std::string Rtcm::print_MSM_6 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM6 (Full GNSS pseudoranges and phaseranges plus CNR, high resolution).
References print_MSM_6().
Referenced by print_MSM_6().

12.337.3.25 print_MSM_7()

```
std::string Rtcm::print_MSM_7 (
            const Gps_Ephemeris & gps_eph,
            const Gps_CNAV_Ephemeris & gps_cnav_eph,
            const Galileo_Ephemeris & gal_eph,
            const Glonass_Gnav_Ephemeris & glo_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint32_t ref_id,
            uint32_t clock_steering_indicator,
            uint32_t external_clock_indicator,
```

```
            int32_t smooth_int,
            bool divergence_free,
            bool more_messages)
```

Prints messages of type MSM7 (Full GNSS pseudoranges, phaseranges, phaserange rate and CNR, high resolution).
References print_MSM_7().
Referenced by print_MSM_7().

12.337.3.26 print_MT1001()

```
std::string Rtcm::print_MT1001 (
            const Gps_Ephemeris & gps_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints message type 1001 (L1-Only GPS RTK Observables).
References print_MT1001().
Referenced by print_MT1001().

12.337.3.27 print_MT1002()

```
std::string Rtcm::print_MT1002 (
            const Gps_Ephemeris & gps_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints message type 1002 (Extended L1-Only GPS RTK Observables).
References print_MT1002().
Referenced by print_MT1002().

12.337.3.28 print_MT1003()

```
std::string Rtcm::print_MT1003 (
            const Gps_Ephemeris & ephL1,
            const Gps_CNAV_Ephemeris & ephL2,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints message type 1003 (L1 & L2 GPS RTK Observables).
References print_MT1003().
Referenced by print_MT1003().

12.337.3.29 print_MT1004()

```
std::string Rtcm::print_MT1004 (
            const Gps_Ephemeris & ephL1,
            const Gps_CNAV_Ephemeris & ephL2,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints message type 1004 (Extended L1 & L2 GPS RTK Observables).
References print_MT1004().
Referenced by print_MT1004().

12.337.3.30 print_MT1005()

```
std::string Rtcm::print_MT1005 (
            uint32_t ref_id,
            double ecef_x,
```

```
            double ecef_y,
            double ecef_z,
            bool gps,
            bool glonass,
            bool galileo,
            bool non_physical,
            bool single_oscillator,
            uint32_t quarter_cycle_indicator)
```

Prints message type 1005 (Stationary Antenna Reference Point).
References print_MT1005().
Referenced by print_MT1005().

12.337.3.31 print_MT1005_test()

```
std::string Rtcm::print_MT1005_test ()
```

For testing purposes.
References print_MT1005_test().
Referenced by print_MT1005_test().

12.337.3.32 print_MT1006()

```
std::string Rtcm::print_MT1006 (
            uint32_t ref_id,
            double ecef_x,
            double ecef_y,
            double ecef_z,
            bool gps,
            bool glonass,
            bool galileo,
            bool non_physical,
            bool single_oscillator,
            uint32_t quarter_cycle_indicator,
            double height)
```

Prints message type 1006 (Stationary Antenna Reference Point, with Height Information).
References print_MT1006().
Referenced by print_MT1006().

12.337.3.33 print_MT1008()

```
std::string Rtcm::print_MT1008 (
            uint32_t ref_id,
            const std::string & antenna_descriptor,
            uint32_t antenna_setup_id,
            const std::string & antenna_serial_number)
```

Prints message type 1008 (Antenna Descriptor & Serial Number).
References print_MT1008().
Referenced by print_MT1008().

12.337.3.34 print_MT1009()

```
std::string Rtcm::print_MT1009 (
            const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints L1-Only GLONASS RTK Observables.
This GLONASS message type is not generally used or supported; type 1012 is to be preferred.

Note

Code added as part of GSoC 2017 program

Parameters

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
obs_time	Time of observation at the moment of printing
observables	Set of observables as defined by the platform

Returns

string with message contents

References print_MT1009().
Referenced by print_MT1009().

12.337.3.35 print_MT1010()

```
std::string Rtcm::print_MT1010 (
            const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints Extended L1-Only GLONASS RTK Observables.
This GLONASS message type is used when only L1 data is present and bandwidth is very tight, often 1012 is used in such cases.

Note

Code added as part of GSoC 2017 program

Parameters

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
obs_time	Time of observation at the moment of printing
observables	Set of observables as defined by the platform

Returns

string with message contents

References print_MT1010().
Referenced by print_MT1010().

12.337.3.36 print_MT1011()

```
std::string Rtcm::print_MT1011 (
            const Glonass_Gnav_Ephemeris & glonass_gnav_ephL1,
            const Glonass_Gnav_Ephemeris & glonass_gnav_ephL2,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints L1&L2 GLONASS RTK Observables.
This GLONASS message type is not generally used or supported; type 1012 is to be preferred

Note

Code added as part of GSoC 2017 program

Parameters

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
obs_time	Time of observation at the moment of printing
observables	Set of observables as defined by the platform

Returns

string with message contents

References print_MT1011().
Referenced by print_MT1011().

12.337.3.37 print_MT1012()

```
std::string Rtcm::print_MT1012 (
            const Glonass_Gnav_Ephemeris & glonass_gnav_ephL1,
            const Glonass_Gnav_Ephemeris & glonass_gnav_ephL2,
            double obs_time,
            const std::map< int32_t, Gnss_Synchro > & observables,
            uint16_t station_id)
```

Prints Extended L1&L2 GLONASS RTK Observables.
This GLONASS message type is the most common observational message type, with L1/L2/SNR content. This is one of the most common messages found.

Note

Code added as part of GSoC 2017 program

Parameters

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
obs_time	Time of observation at the moment of printing
observables	Set of observables as defined by the platform

Returns

string with message contents

References print_MT1012().
Referenced by print_MT1012().

12.337.3.38 print_MT1019()

```
std::string Rtcm::print_MT1019 (
            const Gps_Ephemeris & gps_eph)
```

Prints message type 1019 (GPS Ephemeris), should be broadcast in the event that the IODC does not match the IODE, and every 2 minutes.
References print_MT1019().
Referenced by print_MT1019().

12.337.3.39 print_MT1020()

```
std::string Rtcm::print_MT1020 (
            const Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            const Glonass_Gnav_Utc_Model & glonass_gnav_utc_model)
```

Prints message type 1020 (GLONASS Ephemeris).

Note

Code added as part of GSoC 2017 program

Parameters

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
glonass_gnav_utc_model	GLONASS GNAV Clock Information

Returns

Returns message type as a string type

References print_MT1020().
Referenced by print_MT1020().

12.337.3.40 print_MT1029()

```
std::string Rtcm::print_MT1029 (
            uint32_t ref_id,
            const Gps_Ephemeris & gps_eph,
            double obs_time,
            const std::string & message)
```

Prints message type 1029 (Unicode Text String).
References print_MT1029().
Referenced by print_MT1029().

12.337.3.41 print_MT1045()

```
std::string Rtcm::print_MT1045 (
            const Galileo_Ephemeris & gal_eph)
```

Prints message type 1045 (Galileo Ephemeris), should be broadcast every 2 minutes.
References print_MT1045().
Referenced by print_MT1045().

12.337.3.42 print_MT1057()

```
std::string Rtcm::print_MT1057 (
            const Galileo_HAS_data & has_data,
            bool ssr_multiple_msg_indicator = false)
```

Prints message type 1057 (SSR GPS Orbit Correction).
References print_MT1057().
Referenced by print_MT1057().

12.337.3.43 print_MT1058()

```
std::string Rtcm::print_MT1058 (
            const Galileo_HAS_data & has_data,
            bool use_clock_subset = false,
            bool ssr_multiple_msg_indicator = false)
```

Prints message type 1058 (SSR GPS Clock Correction).
References print_MT1058().
Referenced by print_MT1058().

12.337.3.44 print_MT1059()

```
std::string Rtcm::print_MT1059 (
            const Galileo_HAS_data & has_data,
            bool ssr_multiple_msg_indicator = false)
```

Prints message type 1059 (SSR GPS Satellite Code Bias).
References print_MT1059().
Referenced by print_MT1059().

12.337.3.45 print_MT1060()

```
std::string Rtcm::print_MT1060 (
            const Galileo_HAS_data & has_data,
            bool ssr_multiple_msg_indicator = false)
```

Prints message type 1060 (SSR GPS Combined Orbit and Clock Correction).
References print_MT1060().
Referenced by print_MT1060().

12.337.3.46 read_MT1005()

```
int32_t Rtcm::read_MT1005 (
            const std::string & message,
            uint32_t & ref_id,
            double & ecef_x,
            double & ecef_y,
            double & ecef_z,
            bool & gps,
            bool & glonass,
            bool & galileo)
```

Verifies and reads messages of type 1005 (Stationary Antenna Reference Point). Returns 1 if anything goes wrong, 0 otherwise.
References read_MT1005().
Referenced by read_MT1005().

12.337.3.47 read_MT1019()

```
int32_t Rtcm::read_MT1019 (
            const std::string & message,
            Gps_Ephemeris & gps_eph) const
```

Verifies and reads messages of type 1019 (GPS Ephemeris). Returns 1 if anything goes wrong, 0 otherwise.
References read_MT1019().
Referenced by read_MT1019().

12.337.3.48 read_MT1020()

```
int32_t Rtcm::read_MT1020 (
            const std::string & message,
            Glonass_Gnav_Ephemeris & glonass_gnav_eph,
            Glonass_Gnav_Utc_Model & glonass_gnav_utc_model) const
```

Verifies and reads messages of type 1020 (GLONASS Ephemeris).

Note

Code added as part of GSoC 2017 program

Parameters

message	Message to read as a string type

glonass_gnav_eph	GLONASS GNAV Broadcast Ephemeris
glonass_gnav_utc_model	GLONASS GNAV Clock Information

Returns

Returns 1 if anything goes wrong, 0 otherwise.

References read_MT1020().
Referenced by read_MT1020().

12.337.3.49 read_MT1045()

```
int32_t Rtcm::read_MT1045 (
            const std::string & message,
            Galileo_Ephemeris & gal_eph) const
```

Verifies and reads messages of type 1045 (Galileo Ephemeris). Returns 1 if anything goes wrong, 0 otherwise.
References read_MT1045().
Referenced by read_MT1045().

12.337.3.50 run_server()

```
void Rtcm::run_server ()
```

Starts running the server.
References run_server().
Referenced by run_server().

12.337.3.51 send_message()

```
void Rtcm::send_message (
            const std::string & msg)
```

Sends a message through the server to all connected clients.
References send_message().
Referenced by send_message().

12.337.3.52 stop_server()

```
void Rtcm::stop_server ()
```

Stops the server.
References stop_server().
Referenced by stop_server().
The documentation for this class was generated from the following file:

- rtcm.h

12.338 Rtcm_Printer Class Reference

This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.
`#include <rtcm_printer.h>`

Public Member Functions

- Rtcm_Printer (const std::string &filename, bool flag_rtcm_file_dump, bool flag_rtcm_server, bool flag_rtcm_tty_port, uint16_t rtcm_tcp_port, uint16_t rtcm_station_id, const std::string &rtcm_dump_devname, uint32_t signal_enabled_flags, bool time_tag_name=true, const std::string &base_path=".")

  Default constructor.
- ~Rtcm_Printer () noexcept

  Default destructor.

- void Print_Rtcm_Messages (const Rtklib_Solver *pvt_solver, const std::map< int, Gnss_Synchro > &gnss_observables_map, double rx_time, bool rtcm_MSM_enabled, bool rtcm_MT1019_enabled, bool rtcm_MT1020_enabled, bool rtcm_MT1045_enabled, bool rtcm_MT1077_enabled, bool rtcm_MT1087_enabled, bool rtcm_MT1097_enabled, bool flag_write_RTCM_MSM_output, bool flag_write_RTCM_1019_output, bool flag_write_RTCM_1020_output, bool flag_write_RTCM_1045_output, bool enable_rx_clock_correction)

  Print RTCM messages.
- uint32_t **lock_time** (const Gps_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)
- uint32_t **lock_time** (const Gps_CNAV_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)
- uint32_t **lock_time** (const Galileo_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)
- uint32_t lock_time (const Glonass_Gnav_Ephemeris &eph, double obs_time, const Gnss_Synchro &gnss_synchro)

  Locks time for logging given GLONASS GNAV Broadcast Ephemeris.
- void **Print_IGM_Messages** (const Galileo_HAS_data &has_data)
- void **Print_SSR_Messages** (const Galileo_HAS_data &has_data)
- std::string print_MT1005_test ()

  For testing purposes.

12.338.1 Detailed Description

This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.
Definition at line 49 of file rtcm_printer.h.

12.338.2 Constructor & Destructor Documentation

12.338.2.1 Rtcm_Printer()

```
Rtcm_Printer::Rtcm_Printer (
            const std::string & filename,
            bool flag_rtcm_file_dump,
            bool flag_rtcm_server,
            bool flag_rtcm_tty_port,
            uint16_t rtcm_tcp_port,
            uint16_t rtcm_station_id,
            const std::string & rtcm_dump_devname,
            uint32_t signal_enabled_flags,
            bool time_tag_name = true,
            const std::string & base_path = ".")
```

Default constructor.

12.338.2.2 ~Rtcm_Printer()

```
Rtcm_Printer::~Rtcm_Printer ()  [noexcept]
```

Default destructor.

12.338.3 Member Function Documentation

12.338.3.1 lock_time()

```
uint32_t Rtcm_Printer::lock_time (
            const Glonass_Gnav_Ephemeris & eph,
            double obs_time,
            const Gnss_Synchro & gnss_synchro)
```

Locks time for logging given GLONASS GNAV Broadcast Ephemeris.

Note

Code added as part of GSoC 2017 program \params glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris \params obs_time Time of observation at the moment of printing \params observables Set of observables as defined by the platform

Returns

locked time during logging process

12.338.3.2 print_MT1005_test()

```
std::string Rtcm_Printer::print_MT1005_test ()
```

For testing purposes.
References print_MT1005_test().
Referenced by print_MT1005_test().

12.338.3.3 Print_Rtcm_Messages()

```
void Rtcm_Printer::Print_Rtcm_Messages (
            const Rtklib_Solver * pvt_solver,
            const std::map< int, Gnss_Synchro > & gnss_observables_map,
            double rx_time,
            bool rtcm_MSM_enabled,
            bool rtcm_MT1019_enabled,
            bool rtcm_MT1020_enabled,
            bool rtcm_MT1045_enabled,
            bool rtcm_MT1077_enabled,
            bool rtcm_MT1087_enabled,
            bool rtcm_MT1097_enabled,
            bool flag_write_RTCM_MSM_output,
            bool flag_write_RTCM_1019_output,
            bool flag_write_RTCM_1020_output,
            bool flag_write_RTCM_1045_output,
            bool enable_rx_clock_correction)
```

Print RTCM messages.
References Print_Rtcm_Messages().
Referenced by Print_Rtcm_Messages().
The documentation for this class was generated from the following file:

- rtcm_printer.h

12.339 rtcm_t Struct Reference

Public Attributes

- int staid
- int stah
- int seqno
- int outtype
- gtime_t time
- gtime_t time_s
- obs_t obs
- nav_t nav
- sta_t sta
- dgps_t * dgps
- ssr_t ssr [MAXSAT]
- char msg [128]

- char msgtype [256]
- char msmtype [6][128]
- int obsflag
- int ephsat
- double cp [MAXSAT][NFREQ+NEXOBS]
- unsigned short lock [MAXSAT][NFREQ+NEXOBS]
- unsigned short loss [MAXSAT][NFREQ+NEXOBS]
- gtime_t lltime [MAXSAT][NFREQ+NEXOBS]
- int nbyte
- int nbit
- int len
- unsigned char buff [1200]
- unsigned int word
- unsigned int nmsg2 [100]
- unsigned int nmsg3 [400]
- char opt [256]

12.339.1 Detailed Description

Definition at line 914 of file rtklib.h.

12.339.2 Member Data Documentation

12.339.2.1 buff

```
unsigned char rtcm_t::buff[1200]
```

Definition at line 939 of file rtklib.h.

12.339.2.2 cp

```
double rtcm_t::cp[MAXSAT][NFREQ+NEXOBS]
```

Definition at line 932 of file rtklib.h.

12.339.2.3 dgps

```
dgps_t* rtcm_t::dgps
```

Definition at line 925 of file rtklib.h.

12.339.2.4 ephsat

```
int rtcm_t::ephsat
```

Definition at line 931 of file rtklib.h.

12.339.2.5 len

```
int rtcm_t::len
```

Definition at line 938 of file rtklib.h.

12.339.2.6 lltime

```
gtime_t rtcm_t::lltime[MAXSAT][NFREQ+NEXOBS]
```

Definition at line 935 of file rtklib.h.

12.339.2.7 lock

```
unsigned short rtcm_t::lock[MAXSAT][NFREQ+NEXOBS]
```

Definition at line 933 of file rtklib.h.

12.339.2.8 loss

```
unsigned short rtcm_t::loss[MAXSAT][NFREQ+NEXOBS]
```
Definition at line 934 of file rtklib.h.

12.339.2.9 msg

```
char rtcm_t::msg[128]
```
Definition at line 927 of file rtklib.h.

12.339.2.10 msgtype

```
char rtcm_t::msgtype[256]
```
Definition at line 928 of file rtklib.h.

12.339.2.11 msmtype

```
char rtcm_t::msmtype[6][128]
```
Definition at line 929 of file rtklib.h.

12.339.2.12 nav

```
nav_t rtcm_t::nav
```
Definition at line 923 of file rtklib.h.

12.339.2.13 nbit

```
int rtcm_t::nbit
```
Definition at line 937 of file rtklib.h.

12.339.2.14 nbyte

```
int rtcm_t::nbyte
```
Definition at line 936 of file rtklib.h.

12.339.2.15 nmsg2

```
unsigned int rtcm_t::nmsg2[100]
```
Definition at line 941 of file rtklib.h.

12.339.2.16 nmsg3

```
unsigned int rtcm_t::nmsg3[400]
```
Definition at line 942 of file rtklib.h.

12.339.2.17 obs

```
obs_t rtcm_t::obs
```
Definition at line 922 of file rtklib.h.

12.339.2.18 obsflag

```
int rtcm_t::obsflag
```
Definition at line 930 of file rtklib.h.

12.339.2.19 opt

```
char rtcm_t::opt[256]
```
Definition at line 943 of file rtklib.h.

12.339.2.20 outtype

`int rtcm_t::outtype`
Definition at line 919 of file rtklib.h.

12.339.2.21 seqno

`int rtcm_t::seqno`
Definition at line 918 of file rtklib.h.

12.339.2.22 ssr

`ssr_t rtcm_t::ssr[MAXSAT]`
Definition at line 926 of file rtklib.h.

12.339.2.23 sta

`sta_t rtcm_t::sta`
Definition at line 924 of file rtklib.h.

12.339.2.24 stah

`int rtcm_t::stah`
Definition at line 917 of file rtklib.h.

12.339.2.25 staid

`int rtcm_t::staid`
Definition at line 916 of file rtklib.h.

12.339.2.26 time

`gtime_t rtcm_t::time`
Definition at line 920 of file rtklib.h.

12.339.2.27 time_s

`gtime_t rtcm_t::time_s`
Definition at line 921 of file rtklib.h.

12.339.2.28 word

`unsigned int rtcm_t::word`
Definition at line 940 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.340 rtk_t Struct Reference

Public Attributes

- sol_t sol
- double rb [6]
- int nx
- int na
- double tt
- double * x
- double * P

- double * xa
- double * Pa
- int nfix
- int excsat
- int nb_ar
- int holdamb
- ambc_t ambc [MAXSAT]
- ssat_t ssat [MAXSAT]
- int neb
- char errbuf [MAXERRMSG]
- prcopt_t opt

12.340.1 Detailed Description

Definition at line 1129 of file rtklib.h.

12.340.2 Member Data Documentation

12.340.2.1 ambc

```
ambc_t rtk_t::ambc[MAXSAT]
```

Definition at line 1141 of file rtklib.h.

12.340.2.2 errbuf

```
char rtk_t::errbuf[MAXERRMSG]
```

Definition at line 1144 of file rtklib.h.

12.340.2.3 excsat

```
int rtk_t::excsat
```

Definition at line 1138 of file rtklib.h.

12.340.2.4 holdamb

```
int rtk_t::holdamb
```

Definition at line 1140 of file rtklib.h.

12.340.2.5 na

```
int rtk_t::na
```

Definition at line 1133 of file rtklib.h.

12.340.2.6 nb_ar

```
int rtk_t::nb_ar
```

Definition at line 1139 of file rtklib.h.

12.340.2.7 neb

```
int rtk_t::neb
```

Definition at line 1143 of file rtklib.h.

12.340.2.8 nfix

```
int rtk_t::nfix
```

Definition at line 1137 of file rtklib.h.

12.340.2.9 nx

`int rtk_t::nx`
Definition at line 1133 of file rtklib.h.

12.340.2.10 opt

`prcopt_t rtk_t::opt`
Definition at line 1145 of file rtklib.h.

12.340.2.11 P

`double * rtk_t::P`
Definition at line 1135 of file rtklib.h.

12.340.2.12 Pa

`double * rtk_t::Pa`
Definition at line 1136 of file rtklib.h.

12.340.2.13 rb

`double rtk_t::rb[6]`
Definition at line 1132 of file rtklib.h.

12.340.2.14 sol

`sol_t rtk_t::sol`
Definition at line 1131 of file rtklib.h.

12.340.2.15 ssat

`ssat_t rtk_t::ssat[MAXSAT]`
Definition at line 1142 of file rtklib.h.

12.340.2.16 tt

`double rtk_t::tt`
Definition at line 1134 of file rtklib.h.

12.340.2.17 x

`double* rtk_t::x`
Definition at line 1135 of file rtklib.h.

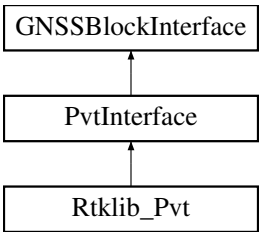
12.340.2.18 xa

`double* rtk_t::xa`
Definition at line 1136 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.341 Rtklib_Pvt Class Reference

This class implements a PvtInterface for the RTKLIB PVT block.
`#include <rtklib_pvt.h>`
Inheritance diagram for Rtklib_Pvt:



Public Member Functions

- **Rtklib_Pvt** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::string role () override
- std::string implementation () override

  Returns "RTKLIB_PVT".
- void clear_ephemeris () override
- std::map< int, Gps_CNAV_Ephemeris > get_gps_cnav_ephemeris () const override
- std::map< int, Glonass_Gnav_Ephemeris > get_glonass_ephemeris () const override
- std::map< int, Glonass_Gnav_Almanac > get_glonass_almanac () const override
- Glonass_Gnav_Utc_Model get_glonass_utc_model () const override
- void clear_ephemeris_keep_almanac () override
- std::map< int, Gps_Ephemeris > get_gps_ephemeris () const override
- std::map< int, Galileo_Ephemeris > get_galileo_ephemeris () const override
- std::map< int, Beidou_Dnav_Ephemeris > get_beidou_dnav_ephemeris () const override
- std::map< int, Beidou_Cnav1_Ephemeris > get_beidou_cnav1_ephemeris () const override
- std::map< int, Beidou_Cnav1_Ephemeris > get_beidou_cnav2_ephemeris () const override
- std::map< int, Gps_Almanac > get_gps_almanac () const override
- std::map< int, Galileo_Almanac > get_galileo_almanac () const override
- std::map< int, Beidou_Dnav_Almanac > get_beidou_dnav_almanac () const override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void reset () override
- size_t item_size () override

  All blocks must have an item_size() function implementation.
- bool get_latest_PVT (double *longitude_deg, double *latitude_deg, double *height_m, double *ground_speed_kmh, double *course_over_ground_deg, time_t *UTC_time) override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.341.1 Detailed Description

This class implements a PvtInterface for the RTKLIB PVT block.
Global configuration options used:
GNSS-SDR.observation_date - approximate date of the signal capture in YYYY-MM-DD or YYYY format, used to resolve the GPS mod-1024 week-number rollover when post-processing recorded files (empty: derive it from the system clock) GNSS-SDR.pre_2009_file - deprecated, use GNSS-SDR.observation_date instead; flag indicating a file captured in the Aug 1999 - Apr 2019 week-number era (false) GNSS-SDR.observable_interval_ms - (20)
It supports the following configuration options:

.dump - (false) .dump_filename - ("./pvt.dat") .dump_mat - (true) .rtk_trace_level - debug level for the RTKLIB methods (0)
.output_rate_ms - (500) Note that the actual rate is the least common multiple of this value and GNSS-SDR.↩ observable_interval_ms .display_rate_ms - (500)
.flag_nmea_tty_port - (false) .nmea_dump_filename - ("./nmea_pvt.nmea") .nmea_dump_devname - ("/dev/tty1")
.rinex_version - (3) values 2 (2.11), 3 (3.02) or 4 (4.02), overridden by -RINEX_version=n.nn command line argument .rinexobs_rate_ms - rate at which RINEX observations are written (1000). Note that the actual rate is the least common multiple of this value and .output_rate_ms .rinex_name - (-RINEX_name command-line argument)
.flag_rtcm_tty_port - (false) .rtcm_dump_devname - ("/dev/pts/1") .flag_rtcm_server - (false) .rtcm_tcp_port - (2101) .rtcm_station_id - (1234) Output rates ... all values are LCM with the computed output rate (above) .rtcm_↩ MT1019_rate_ms - (5000) .rtcm_MT1020_rate_ms - (5000) .rtcm_MT1045_rate_ms - (5000) .rtcm_MSM_rate_ms - (1000) .rtcm_MT1077_rate_ms - (.rtcm_MSM_rate_ms) .rtcm_MT1087_rate_ms - (.rtcm_MSM_rate_ms) .rtcm↩ _MT1097_rate_ms - (.rtcm_MSM_rate_ms)
Fixed-base RTK input through NTRIP (disabled by default): .ntrip_client_enabled - (false) .ntrip_caster_address - caster hostname or IPv4 address, without a scheme or port ("") .ntrip_caster_port - (2101) .ntrip_mountpoint - caster mountpoint ("") .ntrip_username - Basic-authentication username ("") .ntrip_password - Basic-authentication password ("") .ntrip_password_env - environment variable containing the password (""), mutually exclusive with .ntrip_password .ntrip_version - 2 prefers v2 with one compatible v1 retry; 1 forces legacy v1 (2) .ntrip_tls_enabled - TLS 1.2-or-newer transport with system-CA certificate and hostname verification (false) .ntrip_inactivity_timeout_↩ ms - (10000) .ntrip_reconnect_interval_ms - (10000) .ntrip_max_correction_age_s - (5.0) .ntrip_send_gga - upload the rover position as NMEA GGA, required by VRS/nearest-station casters (true) .ntrip_gga_period_ms - period of the GGA upload (10000) .ntrip_station_id - expected RTCM station ID, or zero to accept any stream station (0) .ntrip_fallback_to_single - report SOLQ_SINGLE while fixed-base data is unavailable (true) Supported rover channel sets, per system: GPS 1C alone, 1C+2S, or 1C+L5; Galileo 1B alone or 1B+5X; BeiDou B1C alone; in any combination across systems. num_bands is derived from the channel set, navigation_system must match the enabled constellations, and positioning_mode must be Static or Kinematic. The base must provide its position through RTCM 1005/1006 and observations through legacy 1002/1004 or MSM messages; VRS/nearest-station casters are supported through the periodic GGA upload.
.kml_rate_ms - (1000) .gpx_rate_ms - (1000) .geojson_rate_ms - (1000) .nmea_rate_ms - (1000)
.positioning_mode - The RTKLIB positioning mode. ("Single") Supported values are "Single", "Static", "Kinematic", "PPP_Static" and "PPP_Kinematic". Unsupported modes include DGPS/DGNSS, Moving Baseline, Fixed, and PPP-fixed .num_bands - number of frequencies to use, between 1 and 3. Default is based on the channels configured .elevation_mask - (15.0). Value must be in the range [0,90.0] .dynamics_model - (0) 0:none, 1:velocity, 2:acceleration .satellite_ephemeris - ("Broadcast"). Supported values are "Broadcast" and "SBAS". "SBAS" applies the fast and long-term satellite corrections received from an SBAS telemetry channel. Legacy EGNOS L1 corrections augment GPS L1 observations; satellites without valid corrections are excluded .sbas_satellite - (0). SBAS PRN whose correction stream is used. Zero selects the first received stream; set a PRN in [120,138] when tracking multiple GEOs .iono_model - ("OFF"). Supported values are "OFF", "Broadcast", "SBAS", "Iono-Free-LC", "Estimate_STEC", "IONEX". Unsupported values include QZSS broadcast, QZSS LEX, and SLANT TEC. .trop_↩ model - ("OFF"). Supported values are "OFF", "Saastamoinen", "SBAS", "Estimate_ZTD", and "Estimate_ZTD_↩ Grad". Unsupported values include ZTD correction and ZTD+grad correction .phwindup - phase windup correction for PPP modes (0) .reject_GPS_IIA - whether the GPS Block IIA satellites in eclipse are excluded (0). Only applies in PPP-∗ modes .raim_fde - whether RAIM (receiver autonomous integrity monitoring) FDE (fault detection and exclusion) is enabled (0) .earth_tide - (0) .navigation_system - mask of navigation systems to use. Default based on configured channels 0x01:GPS, 0x02:SBAS, 0x04:GLONASS, 0x08:Galileo, 0x10:QZSS, 0x20:BeiDou, 0x40:IRNS, 0x80:LEO
.AR_GPS - Ambiguity Resolution mode for GPS ("Continuous"). Supported values are "OFF", "Continuous", "↩ Instantaneous", "Fix-and-Hold", "PPP-AR". Unsupported values include PPP-AR ILS, WLNL, and TCAR. .AR_GLO - Ambiguity Resolution mode for GLONASS (1). Value must be in the range [0,3]. (0:off,1:on,2:auto cal,3:ext cal) .AR_DBS - Ambiguity Resolution Mode for BeiDou (1). Value must be in the range [0,1]. (0:off,1:on) .min_ratio_↩ to_fix_ambiguity - (3.0) .min_lock_to_fix_ambiguity - (0) .min_elevation_to_fix_ambiguity - minimum elevation (deg) to fix integer ambiguity (0.0) .outage_reset_ambiguity - (5) .slip_threshold - (0.05) .threshold_reject_gdop - if GDOP is over this value, the observable is excluded (30.0) .threshold_reject_innovation - if innovation is over this value, the observable is excluded (30.0) .number_filter_iter - number of iterations for the estimation filter (1) .bias_0 - (30.0) .iono_0 - (0.03) .trop_0 - (0.3) .sigma_bias - process noise stddev of carrier-phase bias(ambiguity)(cycle/↩ sqrt(s)) (1e-4) .sigma_iono - process noise stddev of vertical ionospheric delay per 10km baseline (m/sqrt(s)) (1e-3) .sigma_trop - process noise stddev of zenith tropospheric delay (m/sqrt(s)) (1e-4) .sigma_acch - process noise stddev of the receiver acceleration horizontal component (m/s2/sqrt(s)) (1e-1) .sigma_accv - process noise stddev

of the receiver acceleration vertical component (m/s2/sqrt(s)) (1e-2) .sigma_pos - (0.0) .code_phase_error_ratio_l1 - (100.0) .code_phase_error_ratio_l2 - (100.0) .code_phase_error_ratio_l5 - (100.0) .carrier_phase_error_factor_a - (0.003) .carrier_phase_error_factor_b - (0.003)
.output_enabled - (true) .rinex_output_enabled - (.output_enabled) .gpx_output_enabled - (.output_enabled) .geojson_output_enabled - (.output_enabled) .kml_output_enabled - (.output_enabled) .xml_output_enabled - (.output_enabled) .nmea_output_enabled - (.output_enabled) .rtcm_output_enabled - (false)
.output_path - directory to which output files are written (".") .rinex_output_path - (.output_path) .gpx_output_↩ path - (.output_path) .geojson_output_path - (.output_path) .kml_output_path - (.output_path) .xml_output_path - (.output_path) .nmea_output_path - (.output_path) .rtcm_output_path - (.output_path)
.enable_monitor - enable the PVT monitor (false) .monitor_client_addresses - ("127.0.0.1") .monitor_udp_port - DO NOT USE THE DEFAULT (1234) .enable_protobuf - serialize using protocol buffers (true). Monitor.enable_protobuf if true, sets this to true
.enable_monitor_ephemeris - enable the ephemeris monitor (false) .monitor_ephemeris_client_addresses - ("127.0.0.1") .monitor_ephemeris_udp_port - DO NOT USE THE DEFAULT (1234)
.show_local_time_zone - (false) .enable_rx_clock_correction - (false) .max_clock_offset_ms - (40)
Definition at line 204 of file rtklib_pvt.h.

12.341.2 Member Function Documentation

12.341.2.1 clear_ephemeris()

```
void Rtklib_Pvt::clear_ephemeris ()  [override], [virtual]
```
Implements PvtInterface.

12.341.2.2 clear_ephemeris_keep_almanac()

```
void Rtklib_Pvt::clear_ephemeris_keep_almanac ()  [override], [virtual]
```
Implements PvtInterface.

12.341.2.3 connect()

```
void Rtklib_Pvt::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Implements GNSSBlockInterface.

12.341.2.4 disconnect()

```
void Rtklib_Pvt::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Implements GNSSBlockInterface.

12.341.2.5 get_beidou_cnav1_ephemeris()

```
std::map< int, Beidou_Cnav1_Ephemeris > Rtklib_Pvt::get_beidou_cnav1_ephemeris () const  [override], [virtual]
```
Reimplemented from PvtInterface.

12.341.2.6 get_beidou_cnav2_ephemeris()

```
std::map< int, Beidou_Cnav1_Ephemeris > Rtklib_Pvt::get_beidou_cnav2_ephemeris () const  [override], [virtual]
```
Reimplemented from PvtInterface.

12.341.2.7 get_beidou_dnav_almanac()

```
std::map< int, Beidou_Dnav_Almanac > Rtklib_Pvt::get_beidou_dnav_almanac () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.8 get_beidou_dnav_ephemeris()

```
std::map< int, Beidou_Dnav_Ephemeris > Rtklib_Pvt::get_beidou_dnav_ephemeris () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.9 get_galileo_almanac()

```
std::map< int, Galileo_Almanac > Rtklib_Pvt::get_galileo_almanac () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.10 get_galileo_ephemeris()

```
std::map< int, Galileo_Ephemeris > Rtklib_Pvt::get_galileo_ephemeris () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.11 get_glonass_almanac()

```
std::map< int, Glonass_Gnav_Almanac > Rtklib_Pvt::get_glonass_almanac () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.12 get_glonass_ephemeris()

```
std::map< int, Glonass_Gnav_Ephemeris > Rtklib_Pvt::get_glonass_ephemeris () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.13 get_glonass_utc_model()

```
Glonass_Gnav_Utc_Model Rtklib_Pvt::get_glonass_utc_model () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.14 get_gps_almanac()

```
std::map< int, Gps_Almanac > Rtklib_Pvt::get_gps_almanac () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.15 get_gps_cnav_ephemeris()

```
std::map< int, Gps_CNAV_Ephemeris > Rtklib_Pvt::get_gps_cnav_ephemeris () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.16 get_gps_ephemeris()

```
std::map< int, Gps_Ephemeris > Rtklib_Pvt::get_gps_ephemeris () const  [override], [virtual]
```
Implements PvtInterface.

12.341.2.17 get_latest_PVT()

```
bool Rtklib_Pvt::get_latest_PVT (
            double * longitude_deg,
            double * latitude_deg,
            double * height_m,
            double * ground_speed_kmh,
            double * course_over_ground_deg,
            time_t * UTC_time)  [override], [virtual]
```
Implements PvtInterface.

12.341.2.18 get_left_block()

`gr::basic_block_sptr Rtklib_Pvt::get_left_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.341.2.19 get_right_block()

`gr::basic_block_sptr Rtklib_Pvt::get_right_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.341.2.20 implementation()

`std::string Rtklib_Pvt::implementation () [inline], [override], [virtual]`
Returns "RTKLIB_PVT".
Implements GNSSBlockInterface.
Definition at line 220 of file rtklib_pvt.h.

12.341.2.21 item_size()

`size_t Rtklib_Pvt::item_size () [inline], [override], [virtual]`
All blocks must have an item_size() function implementation.
Implements GNSSBlockInterface.
Definition at line 251 of file rtklib_pvt.h.

12.341.2.22 reset()

`void Rtklib_Pvt::reset () [inline], [override], [virtual]`
Implements PvtInterface.
Definition at line 245 of file rtklib_pvt.h.

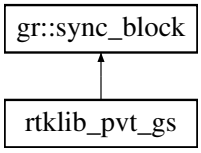
12.341.2.23 role()

`std::string Rtklib_Pvt::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 214 of file rtklib_pvt.h.
The documentation for this class was generated from the following file:

- rtklib_pvt.h

12.342 rtklib_pvt_gs Class Reference

This class implements a block that computes the PVT solution using the RTKLIB integrated library.
`#include <rtklib_pvt_gs.h>`
Inheritance diagram for rtklib_pvt_gs:



Public Member Functions

- ~rtklib_pvt_gs () override
  Default destructor.
- std::map< int, Gps_CNAV_Ephemeris > get_gps_cnav_ephemeris_map () const
  Get latest set of GPS ephemeris from PVT block.
- std::map< int, Glonass_Gnav_Ephemeris > **get_glonass_ephemeris_map** () const

- std::map< int, Glonass_Gnav_Almanac > **get_glonass_almanac_map** () const
- Glonass_Gnav_Utc_Model **get_glonass_utc_model** () const
- std::map< int, Gps_Ephemeris > **get_gps_ephemeris_map** () const
- std::map< int, Gps_Almanac > get_gps_almanac_map () const

  Get latest set of GPS almanac from PVT block.
- std::map< int, Galileo_Ephemeris > get_galileo_ephemeris_map () const

  Get latest set of Galileo ephemeris from PVT block.
- std::map< int, Galileo_Almanac > get_galileo_almanac_map () const

  Get latest set of Galileo almanac from PVT block.
- std::map< int, Beidou_Dnav_Ephemeris > get_beidou_dnav_ephemeris_map () const

  Get latest set of BeiDou DNAV ephemeris from PVT block.
- std::map< int, Beidou_Cnav1_Ephemeris > **get_beidou_cnav1_ephemeris_map** () const
- std::map< int, Beidou_Cnav1_Ephemeris > **get_beidou_cnav2_ephemeris_map** () const
- std::map< int, Beidou_Dnav_Almanac > get_beidou_dnav_almanac_map () const

  Get latest set of BeiDou DNAV almanac from PVT block.
- void clear_ephemeris ()

  Clears the published navigation snapshot now and the solver maps at the worker's next work() or telemetry callback.
- void clear_ephemeris_keep_almanac ()

  Clears the ephemeris entries of the published navigation snapshot now and the solver ephemeris maps at the worker's next work() or telemetry callback, leaving the GPS, Galileo, BeiDou and GLONASS almanacs untouched.
- bool get_latest_PVT (double *longitude_deg, double *latitude_deg, double *height_m, double *ground_speed_kmh, double *course_over_ground_deg, time_t *UTC_time) const

  Get the latest Position WGS84 [deg], Ground Velocity, Course over Ground, and UTC Time, if available.
- int work (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  PVT Signal Processing.

Static Public Member Functions

- static Gnss_Synchro interpolate_observable (const Gnss_Synchro &early, const Gnss_Synchro &late, double time_factor, double rx_time_s)

  Interpolates one observable between two epochs, in the carrier polarity frame of the later epoch.

Friends

- rtklib_pvt_gs_sptr **rtklib_make_pvt_gs** (uint32_t nchannels, const Pvt_Conf &conf_, const rtk_t &rtk, const SensorDataSourceConfiguration &sensor_data_configuration)

12.342.1 Detailed Description

This class implements a block that computes the PVT solution using the RTKLIB integrated library.
Definition at line 95 of file rtklib_pvt_gs.h.

12.342.2 Constructor & Destructor Documentation

12.342.2.1 ~rtklib_pvt_gs()

```
rtklib_pvt_gs::~rtklib_pvt_gs () [override]
```

Default destructor.

12.342.3 Member Function Documentation

12.342.3.1 clear_ephemeris()

```
void rtklib_pvt_gs::clear_ephemeris ()
```

Clears the published navigation snapshot now and the solver maps at the worker's next work() or telemetry callback.

12.342.3.2 clear_ephemeris_keep_almanac()

```
void rtklib_pvt_gs::clear_ephemeris_keep_almanac ()
```

Clears the ephemeris entries of the published navigation snapshot now and the solver ephemeris maps at the worker's next work() or telemetry callback, leaving the GPS, Galileo, BeiDou and GLONASS almanacs untouched.

12.342.3.3 get_beidou_dnav_almanac_map()

```
std::map< int, Beidou_Dnav_Almanac > rtklib_pvt_gs::get_beidou_dnav_almanac_map () const
```

Get latest set of BeiDou DNAV almanac from PVT block.

12.342.3.4 get_beidou_dnav_ephemeris_map()

```
std::map< int, Beidou_Dnav_Ephemeris > rtklib_pvt_gs::get_beidou_dnav_ephemeris_map () const
```

Get latest set of BeiDou DNAV ephemeris from PVT block.

12.342.3.5 get_galileo_almanac_map()

```
std::map< int, Galileo_Almanac > rtklib_pvt_gs::get_galileo_almanac_map () const
```

Get latest set of Galileo almanac from PVT block.

12.342.3.6 get_galileo_ephemeris_map()

```
std::map< int, Galileo_Ephemeris > rtklib_pvt_gs::get_galileo_ephemeris_map () const
```

Get latest set of Galileo ephemeris from PVT block.

12.342.3.7 get_gps_almanac_map()

```
std::map< int, Gps_Almanac > rtklib_pvt_gs::get_gps_almanac_map () const
```

Get latest set of GPS almanac from PVT block.

12.342.3.8 get_gps_cnav_ephemeris_map()

```
std::map< int, Gps_CNAV_Ephemeris > rtklib_pvt_gs::get_gps_cnav_ephemeris_map () const
```

Get latest set of GPS ephemeris from PVT block.

12.342.3.9 get_latest_PVT()

```
bool rtklib_pvt_gs::get_latest_PVT (
            double * longitude_deg,
            double * latitude_deg,
            double * height_m,
            double * ground_speed_kmh,
            double * course_over_ground_deg,
            time_t * UTC_time) const
```

Get the latest Position WGS84 [deg], Ground Velocity, Course over Ground, and UTC Time, if available.

12.342.3.10 interpolate_observable()

```
Gnss_Synchro rtklib_pvt_gs::interpolate_observable (
            const Gnss_Synchro & early,
            const Gnss_Synchro & late,
            double time_factor,
            double rx_time_s)  [static]
```

Interpolates one observable between two epochs, in the carrier polarity frame of the later epoch.
A plain linear carrier-phase interpolation across a PLL-180 half-cycle re-resolution would leak a fraction of the pi step into the output with no slip flag attached; the early phase is therefore shifted into the late epoch's polarity frame first, the late polarity is stamped on the output, and a slip on either endpoint (or a polarity change itself) marks the output.

12.342.3.11 work()

```
int rtklib_pvt_gs::work (
            int noutput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

PVT Signal Processing.

The documentation for this class was generated from the following file:

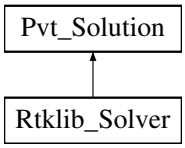
- rtklib_pvt_gs.h

12.343 Rtklib_Solver Class Reference

This class implements a PVT solution based on RTKLIB.

`#include <rtklib_solver.h>`

Inheritance diagram for Rtklib_Solver:



Public Member Functions

- **Rtklib_Solver** (const rtk_t &rtk, const Pvt_Conf &conf, const std::string &dump_filename, uint32_t signal_↩ enabled_flags, bool flag_dump_to_file, bool flag_dump_to_mat)
- **Rtklib_Solver** (const Rtklib_Solver &)=delete
- Rtklib_Solver & **operator=** (const Rtklib_Solver &)=delete
- **Rtklib_Solver** (Rtklib_Solver &&)=delete
- Rtklib_Solver & **operator=** (Rtklib_Solver &&)=delete
- bool **get_PVT** (const std::map< int, Gnss_Synchro > &gnss_observables_map, double kf_update_↩ interval_s, const SensorDataAggregator &sensor_data_aggregator, bool dump_this_epoch=true, const Ntrip_Rtcm_Snapshot *fixed_base=nullptr)
- double get_hdop () const override
- double get_vdop () const override
- double get_pdop () const override
- double get_gdop () const override
- Monitor_Pvt **get_monitor_pvt** () const
- Rtklib_Fixed_Base_Status **get_fixed_base_status** () const
- double **get_fixed_base_age_s** () const
- std::size_t **get_fixed_base_common_satellites** () const
- bool has_precise_ephemeris () const

  true only if precise ephemeris and clock products are loaded; used to distinguish real PPP from PPP-mode processing with broadcast products
- bool solution_attempted () const

  true if the last call to get_PVT() had what it takes to compute a solution, whether it succeeded or not; false if it returned without trying (fewer than four satellites with ephemeris, or no fixed-base data with the fallback to single-point positioning disabled)
- void **store_has_data** (const Galileo_HAS_data &new_has_data)
- void **clear_has_corrections** ()
- void **update_has_corrections** (const std::map< int, Gnss_Synchro > &obs_map)
- void **store_sbas_message** (const Sbas_Raw_Message &message)
- void **store_gps_ephemeris** (const Gps_Ephemeris &ephemeris)
- void **clear_gps_ephemerides** ()
- bool **store_galileo_ephemeris** (const Galileo_Ephemeris &ephemeris)

- Galileo_Nav_Message_Type **galileo_nav_message_type_for_pvt** () const
- bool **is_galileo_signal_used_in_pvt** (const std::string &signal) const
- bool **get_galileo_signal_health** (uint32_t prn, const std::string &signal, uint32_t observation_tow, bool &healthy) const
- bool get_broadcast_signal_health (char system, uint32_t prn, const std::string &signal, uint32_t observation_tow, bool &healthy) const

  Broadcast health of one tracked signal, as reported by the navigation message that carries it: GPS/QZSS L1 C/A from the LNAV SV health (almanac when no ephemeris has been decoded yet), GPS L2C and QZSS L5 from the CNAV per-signal health bits, GPS L5 from the LNAV SV health of the same satellite (the CNAV L5 bit is broadcast as unhealthy while GPS L5 is pre-operational), Galileo per signal (see get_galileo_signal_health(), the almanac health status of the same signal when no ephemeris is available), GLONASS from the GNAV ln/Bn flags, BeiDou B1I/B3I from the DNAV SV health and B1C/B2a from the B-CNAV1/B-CNAV2 health status. Independent of PVT.use_unhealthy_sats.
- std::map< int, Galileo_Ephemeris > **get_galileo_ephemeris_map_for_pvt** () const
- bool **select_galileo_ephemeris** (uint32_t prn, const std::string &signal, uint32_t observation_tow, Galileo_Ephemeris &ephemeris, bool &from_reduced_ced) const

Public Member Functions inherited from Pvt_Solution

- std::array< double, 3 > **get_rx_pos** () const
- std::array< double, 3 > **get_rx_vel** () const
- boost::posix_time::ptime **get_position_UTC_time** () const
- double get_latitude () const

  Get RX position Latitude WGS84 [deg].
- double get_longitude () const

  Get RX position Longitude WGS84 [deg].
- double get_height () const

  Get RX position height WGS84 [m].
- double get_time_offset_s () const

  Get RX time offset [s].
- double get_clock_drift_ppm () const

  Get the Rx clock drift [ppm].
- double get_speed_over_ground () const

  Get RX speed over ground [m/s].
- double get_course_over_ground () const

  Get RX course over ground [deg].
- int get_num_valid_observations () const

  Get the number of valid pseudorange observations (valid satellites).
- int32_t get_ref_gps_week () const

  Get the reference GPS week used to resolve the mod-1024 week rollover (0 if unset).
- bool **is_valid_position** () const
- void set_rx_pos (const std::array< double, 3 > &pos)

  Set position: X, Y, Z in Cartesian ECEF coordinates [m].
- void set_rx_vel (const std::array< double, 3 > &vel)

  Set velocity: East [m/s], North [m/s], Up [m/s].
- void **set_position_UTC_time** (const boost::posix_time::ptime &pt)
- void set_time_offset_s (double offset)

  Set RX time offset [s].
- void set_clock_drift_ppm (double clock_drift_ppm)

  Set the Rx clock drift [ppm].
- void set_speed_over_ground (double speed_m_s)

  Set RX speed over ground [m/s].
- void set_course_over_ground (double cog_deg)

Set RX course over ground [deg].

- void **set_valid_position** (bool is_valid)
- void set_num_valid_observations (int num)

  Set the number of valid pseudorange observations (valid satellites).
- void set_ref_gps_week (int32_t ref_week)

  Set the reference GPS week (e.g., derived from the date of signal capture) used to resolve the mod-1024 week rollover in post-processing mode.

Public Attributes

- sol_t pvt_sol {}
- std::array< ssat_t, MAXSAT > pvt_ssat {}
- Galileo_Ephemeris_Store galileo_ephemeris_store

  Source-aware Galileo ephemeris storage.
- std::map< int, Galileo_Ephemeris > galileo_ephemeris_map

  Compatibility PVT view; source-aware storage is authoritative.
- std::map< int, Galileo_Reduced_CED > galileo_reduced_ced_map

  Map storing provisional Galileo Reduced CED.
- std::map< int, Gps_Ephemeris > gps_ephemeris_map

  Map storing new GPS_Ephemeris.
- std::map< int, Gps_Ephemeris > gps_previous_ephemeris_map

  Previous GPS/QZSS issue retained for SBAS IODE handover.
- std::map< int, Gps_CNAV_Ephemeris > gps_cnav_ephemeris_map

  Map storing new GPS_CNAV_Ephemeris.
- std::map< int, Glonass_Gnav_Ephemeris > glonass_gnav_ephemeris_map

  Map storing new GLONASS GNAV Ephemeris.
- std::map< int, Beidou_Dnav_Ephemeris > beidou_dnav_ephemeris_map

  Map storing new BeiDou DNAV Ephmeris.
- std::map< int, Beidou_Cnav1_Ephemeris > beidou_cnav1_ephemeris_map

  Map storing BeiDou B-CNAV1 ephemeris.
- std::map< int, Beidou_Cnav1_Ephemeris > beidou_cnav2_ephemeris_map

  Map storing BeiDou B-CNAV2 ephemeris.
- Galileo_Utc_Model galileo_utc_model
- Galileo_Iono galileo_iono
- std::map< int, Galileo_Almanac > galileo_almanac_map
- Gps_Utc_Model gps_utc_model
- Gps_Iono gps_iono
- std::map< int, Gps_Almanac > gps_almanac_map
- Gps_CNAV_Eop gps_cnav_eop
- Gps_CNAV_Iono gps_cnav_iono
- Gps_CNAV_Utc_Model gps_cnav_utc_model
- Qzss_Utc_Model qzss_utc_model
- Qzss_Iono qzss_iono
- Qzss_CNAV_Eop qzss_cnav_eop
- Qzss_CNAV_Iono qzss_cnav_iono
- Qzss_CNAV_Utc_Model qzss_cnav_utc_model
- Glonass_Gnav_Utc_Model glonass_gnav_utc_model

  Map storing GLONASS GNAV UTC Model.
- std::map< int, Glonass_Gnav_Almanac > glonass_gnav_almanac_map
- Glonass_Gnav_Almanac glonass_gnav_almanac

  Map storing GLONASS GNAV Almanac Model.
- Beidou_Dnav_Utc_Model beidou_dnav_utc_model
- Beidou_Dnav_Iono beidou_dnav_iono

- Beidou_Cnav1_Iono beidou_cnav1_iono
- Beidou_Cnav1_Utc_Model beidou_cnav1_utc_model
- std::map< int, Beidou_Dnav_Almanac > beidou_dnav_almanac_map
- std::map< int, Bds3_B1c_PageData > beidou_cnav1_page_data_map

Static Public Attributes

- static constexpr int NTRIP_MIN_COMMON_SATELLITES = 4

  Minimum satellites common to rover and base (any constellation) for the fixed-base merge; the solver-owned policy the display layer quotes.

12.343.1 Detailed Description

This class implements a PVT solution based on RTKLIB.
Definition at line 112 of file rtklib_solver.h.

12.343.2 Member Function Documentation

12.343.2.1 get_broadcast_signal_health()

```
bool Rtklib_Solver::get_broadcast_signal_health (
            char system,
            uint32_t prn,
            const std::string & signal,
            uint32_t observation_tow,
            bool & healthy) const
```

Broadcast health of one tracked signal, as reported by the navigation message that carries it: GPS/QZSS L1 C/A from the LNAV SV health (almanac when no ephemeris has been decoded yet), GPS L2C and QZSS L5 from the CNAV per-signal health bits, GPS L5 from the LNAV SV health of the same satellite (the CNAV L5 bit is broadcast as unhealthy while GPS L5 is pre-operational), Galileo per signal (see get_galileo_signal_health(), the almanac health status of the same signal when no ephemeris is available), GLONASS from the GNAV ln/Bn flags, Bei↩ Dou B1I/B3I from the DNAV SV health and B1C/B2a from the B-CNAV1/B-CNAV2 health status. Independent of PVT.use_unhealthy_sats.

Returns

true if health information is available for that signal (healthy is then set)

12.343.2.2 get_gdop()

```
double Rtklib_Solver::get_gdop () const  [override], [virtual]
```

Implements Pvt_Solution.

12.343.2.3 get_hdop()

```
double Rtklib_Solver::get_hdop () const  [override], [virtual]
```

Implements Pvt_Solution.

12.343.2.4 get_pdop()

```
double Rtklib_Solver::get_pdop () const  [override], [virtual]
```

Implements Pvt_Solution.

12.343.2.5 get_vdop()

```
double Rtklib_Solver::get_vdop () const  [override], [virtual]
```

Implements Pvt_Solution.

12.343.2.6 has_precise_ephemeris()

```
bool Rtklib_Solver::has_precise_ephemeris () const  [inline]
```

true only if precise ephemeris and clock products are loaded; used to distinguish real PPP from PPP-mode processing with broadcast products
Definition at line 150 of file rtklib_solver.h.
References has_precise_ephemeris().
Referenced by has_precise_ephemeris().

12.343.2.7 solution_attempted()

```
bool Rtklib_Solver::solution_attempted () const  [inline]
```

true if the last call to get_PVT() had what it takes to compute a solution, whether it succeeded or not; false if it returned without trying (fewer than four satellites with ephemeris, or no fixed-base data with the fallback to single-point positioning disabled)
Definition at line 159 of file rtklib_solver.h.

12.343.3 Member Data Documentation

12.343.3.1 beidou_cnav1_ephemeris_map

```
std::map<int, Beidou_Cnav1_Ephemeris> Rtklib_Solver::beidou_cnav1_ephemeris_map
```

Map storing BeiDou B-CNAV1 ephemeris.
Definition at line 205 of file rtklib_solver.h.

12.343.3.2 beidou_cnav1_iono

```
Beidou_Cnav1_Iono Rtklib_Solver::beidou_cnav1_iono
```

Definition at line 233 of file rtklib_solver.h.

12.343.3.3 beidou_cnav1_page_data_map

```
std::map<int, Bds3_B1c_PageData> Rtklib_Solver::beidou_cnav1_page_data_map
```

Definition at line 236 of file rtklib_solver.h.

12.343.3.4 beidou_cnav1_utc_model

```
Beidou_Cnav1_Utc_Model Rtklib_Solver::beidou_cnav1_utc_model
```

Definition at line 234 of file rtklib_solver.h.

12.343.3.5 beidou_cnav2_ephemeris_map

```
std::map<int, Beidou_Cnav1_Ephemeris> Rtklib_Solver::beidou_cnav2_ephemeris_map
```

Map storing BeiDou B-CNAV2 ephemeris.
Definition at line 206 of file rtklib_solver.h.

12.343.3.6 beidou_dnav_almanac_map

```
std::map<int, Beidou_Dnav_Almanac> Rtklib_Solver::beidou_dnav_almanac_map
```

Definition at line 235 of file rtklib_solver.h.

12.343.3.7 beidou_dnav_ephemeris_map

```
std::map<int, Beidou_Dnav_Ephemeris> Rtklib_Solver::beidou_dnav_ephemeris_map
```

Map storing new BeiDou DNAV Ephmeris.
Definition at line 204 of file rtklib_solver.h.

12.343.3.8 beidou_dnav_iono

```
Beidou_Dnav_Iono Rtklib_Solver::beidou_dnav_iono
```

Definition at line 232 of file rtklib_solver.h.

12.343.3.9 beidou_dnav_utc_model

Beidou_Dnav_Utc_Model Rtklib_Solver::beidou_dnav_utc_model
Definition at line 231 of file rtklib_solver.h.

12.343.3.10 galileo_almanac_map

std::map<int, Galileo_Almanac> Rtklib_Solver::galileo_almanac_map
Definition at line 210 of file rtklib_solver.h.

12.343.3.11 galileo_ephemeris_map

std::map<int, Galileo_Ephemeris> Rtklib_Solver::galileo_ephemeris_map
Compatibility PVT view; source-aware storage is authoritative.
Definition at line 198 of file rtklib_solver.h.

12.343.3.12 galileo_ephemeris_store

Galileo_Ephemeris_Store Rtklib_Solver::galileo_ephemeris_store
Source-aware Galileo ephemeris storage.
Definition at line 197 of file rtklib_solver.h.

12.343.3.13 galileo_iono

Galileo_Iono Rtklib_Solver::galileo_iono
Definition at line 209 of file rtklib_solver.h.

12.343.3.14 galileo_reduced_ced_map

std::map<int, Galileo_Reduced_CED> Rtklib_Solver::galileo_reduced_ced_map
Map storing provisional Galileo Reduced CED.
Definition at line 199 of file rtklib_solver.h.

12.343.3.15 galileo_utc_model

Galileo_Utc_Model Rtklib_Solver::galileo_utc_model
Definition at line 208 of file rtklib_solver.h.

12.343.3.16 glonass_gnav_almanac

Glonass_Gnav_Almanac Rtklib_Solver::glonass_gnav_almanac
Map storing GLONASS GNAV Almanac Model.
Definition at line 229 of file rtklib_solver.h.

12.343.3.17 glonass_gnav_almanac_map

std::map<int, Glonass_Gnav_Almanac> Rtklib_Solver::glonass_gnav_almanac_map
Definition at line 228 of file rtklib_solver.h.

12.343.3.18 glonass_gnav_ephemeris_map

std::map<int, Glonass_Gnav_Ephemeris> Rtklib_Solver::glonass_gnav_ephemeris_map
Map storing new GLONASS GNAV Ephemeris.
Definition at line 203 of file rtklib_solver.h.

12.343.3.19 glonass_gnav_utc_model

Glonass_Gnav_Utc_Model Rtklib_Solver::glonass_gnav_utc_model
Map storing GLONASS GNAV UTC Model.
Definition at line 227 of file rtklib_solver.h.

12.343.3.20 gps_almanac_map

`std::map<int, Gps_Almanac> Rtklib_Solver::gps_almanac_map`
Definition at line 214 of file rtklib_solver.h.

12.343.3.21 gps_cnav_eop

`Gps_CNAV_Eop Rtklib_Solver::gps_cnav_eop`
Definition at line 216 of file rtklib_solver.h.

12.343.3.22 gps_cnav_ephemeris_map

`std::map<int, Gps_CNAV_Ephemeris> Rtklib_Solver::gps_cnav_ephemeris_map`
Map storing new GPS_CNAV_Ephemeris.
Definition at line 202 of file rtklib_solver.h.

12.343.3.23 gps_cnav_iono

`Gps_CNAV_Iono Rtklib_Solver::gps_cnav_iono`
Definition at line 217 of file rtklib_solver.h.

12.343.3.24 gps_cnav_utc_model

`Gps_CNAV_Utc_Model Rtklib_Solver::gps_cnav_utc_model`
Definition at line 218 of file rtklib_solver.h.

12.343.3.25 gps_ephemeris_map

`std::map<int, Gps_Ephemeris> Rtklib_Solver::gps_ephemeris_map`
Map storing new GPS_Ephemeris.
Definition at line 200 of file rtklib_solver.h.

12.343.3.26 gps_iono

`Gps_Iono Rtklib_Solver::gps_iono`
Definition at line 213 of file rtklib_solver.h.

12.343.3.27 gps_previous_ephemeris_map

`std::map<int, Gps_Ephemeris> Rtklib_Solver::gps_previous_ephemeris_map`
Previous GPS/QZSS issue retained for SBAS IODE handover.
Definition at line 201 of file rtklib_solver.h.

12.343.3.28 gps_utc_model

`Gps_Utc_Model Rtklib_Solver::gps_utc_model`
Definition at line 212 of file rtklib_solver.h.

12.343.3.29 NTRIP_MIN_COMMON_SATELLITES

`int Rtklib_Solver::NTRIP_MIN_COMMON_SATELLITES = 4 [static], [constexpr]`
Minimum satellites common to rover and base (any constellation) for the fixed-base merge; the solver-owned policy the display layer quotes.
Definition at line 117 of file rtklib_solver.h.

12.343.3.30 pvt_sol

`sol_t Rtklib_Solver::pvt_sol {}`
Definition at line 194 of file rtklib_solver.h.

12.343.3.31 pvt_ssat

`std::array<ssat_t, MAXSAT> Rtklib_Solver::pvt_ssat {}`
Definition at line 195 of file rtklib_solver.h.

12.343.3.32 qzss_cnav_eop

`Qzss_CNAV_Eop Rtklib_Solver::qzss_cnav_eop`
Definition at line 223 of file rtklib_solver.h.

12.343.3.33 qzss_cnav_iono

`Qzss_CNAV_Iono Rtklib_Solver::qzss_cnav_iono`
Definition at line 224 of file rtklib_solver.h.

12.343.3.34 qzss_cnav_utc_model

`Qzss_CNAV_Utc_Model Rtklib_Solver::qzss_cnav_utc_model`
Definition at line 225 of file rtklib_solver.h.

12.343.3.35 qzss_iono

`Qzss_Iono Rtklib_Solver::qzss_iono`
Definition at line 221 of file rtklib_solver.h.

12.343.3.36 qzss_utc_model

`Qzss_Utc_Model Rtklib_Solver::qzss_utc_model`
Definition at line 220 of file rtklib_solver.h.
The documentation for this class was generated from the following file:

- rtklib_solver.h

12.344 Rtklib_Tls_Client Class Reference

Public Member Functions

- **Rtklib_Tls_Client** (std::string hostname)
- **Rtklib_Tls_Client** (const Rtklib_Tls_Client &)=delete
- Rtklib_Tls_Client & **operator=** (const Rtklib_Tls_Client &)=delete
- **Rtklib_Tls_Client** (Rtklib_Tls_Client &&)=delete
- Rtklib_Tls_Client & **operator=** (Rtklib_Tls_Client &&)=delete
- bool **initialize** (char ∗msg)
- void **reset** ()
- bool **set_hostname** (const std::string &hostname)
- int **handshake** (socket_t sock, char ∗msg)
- int **read** (unsigned char ∗buff, int n, char ∗msg)
- int **write** (const unsigned char ∗buff, int n, char ∗msg)

Friends

- class Rtklib_Tls_Client_Test_Access

12.344.1 Detailed Description

Definition at line 26 of file rtklib_tls.h.

12.344.2 Friends And Related Symbol Documentation

12.344.2.1 Rtklib_Tls_Client_Test_Access

```
friend class Rtklib_Tls_Client_Test_Access  [friend]
```

Definition at line 56 of file rtklib_tls.h.

The documentation for this class was generated from the following file:

- rtklib_tls.h

12.345 rtksvr_t Struct Reference

Public Attributes

- int state
- int cycle
- int nmeacycle
- int nmeareq
- double nmeapos [3]
- int buffsize
- int format [3]
- solopt_t solopt [2]
- int navsel
- int nsbs
- int nsol
- rtk_t rtk
- int nb [3]
- int nsb [2]
- int npb [3]
- unsigned char * buff [3]
- unsigned char * sbuf [2]
- unsigned char * pbuf [3]
- sol_t solbuf [MAXSOLBUF]
- unsigned int nmsg [3][10]
- raw_t raw [3]
- rtcm_t rtcm [3]
- gtime_t ftime [3]
- char files [3][MAXSTRPATH]
- obs_t obs [3][MAXOBSBUF]
- nav_t nav
- sbsmsg_t sbsmsg [MAXSBSMSG]
- stream_t stream [8]
- stream_t * moni
- unsigned int tick
- pthread_t thread
- int cputime
- int prcout
- lock_t lock

12.345.1 Detailed Description

Definition at line 1318 of file rtklib.h.

12.345.2 Member Data Documentation

12.345.2.1 buff

```
unsigned char* rtksvr_t::buff[3]
```

Definition at line 1335 of file rtklib.h.

12.345.2.2 buffsize

```
int rtksvr_t::buffsize
```
Definition at line 1325 of file rtklib.h.

12.345.2.3 cputime

```
int rtksvr_t::cputime
```
Definition at line 1351 of file rtklib.h.

12.345.2.4 cycle

```
int rtksvr_t::cycle
```
Definition at line 1321 of file rtklib.h.

12.345.2.5 files

```
char rtksvr_t::files[3][MAXSTRPATH]
```
Definition at line 1343 of file rtklib.h.

12.345.2.6 format

```
int rtksvr_t::format[3]
```
Definition at line 1326 of file rtklib.h.

12.345.2.7 ftime

```
gtime_t rtksvr_t::ftime[3]
```
Definition at line 1342 of file rtklib.h.

12.345.2.8 lock

```
lock_t rtksvr_t::lock
```
Definition at line 1353 of file rtklib.h.

12.345.2.9 moni

```
stream_t* rtksvr_t::moni
```
Definition at line 1348 of file rtklib.h.

12.345.2.10 nav

```
nav_t rtksvr_t::nav
```
Definition at line 1345 of file rtklib.h.

12.345.2.11 navsel

```
int rtksvr_t::navsel
```
Definition at line 1328 of file rtklib.h.

12.345.2.12 nb

```
int rtksvr_t::nb[3]
```
Definition at line 1332 of file rtklib.h.

12.345.2.13 nmeacycle

```
int rtksvr_t::nmeacycle
```
Definition at line 1322 of file rtklib.h.

12.345.2.14 nmeapos

`double rtksvr_t::nmeapos[3]`
Definition at line 1324 of file rtklib.h.

12.345.2.15 nmeareq

`int rtksvr_t::nmeareq`
Definition at line 1323 of file rtklib.h.

12.345.2.16 nmsg

`unsigned int rtksvr_t::nmsg[3][10]`
Definition at line 1339 of file rtklib.h.

12.345.2.17 npb

`int rtksvr_t::npb[3]`
Definition at line 1334 of file rtklib.h.

12.345.2.18 nsb

`int rtksvr_t::nsb[2]`
Definition at line 1333 of file rtklib.h.

12.345.2.19 nsbs

`int rtksvr_t::nsbs`
Definition at line 1329 of file rtklib.h.

12.345.2.20 nsol

`int rtksvr_t::nsol`
Definition at line 1330 of file rtklib.h.

12.345.2.21 obs

`obs_t rtksvr_t::obs[3][MAXOBSBUF]`
Definition at line 1344 of file rtklib.h.

12.345.2.22 pbuf

`unsigned char* rtksvr_t::pbuf[3]`
Definition at line 1337 of file rtklib.h.

12.345.2.23 prcout

`int rtksvr_t::prcout`
Definition at line 1352 of file rtklib.h.

12.345.2.24 raw

`raw_t rtksvr_t::raw[3]`
Definition at line 1340 of file rtklib.h.

12.345.2.25 rtcm

`rtcm_t rtksvr_t::rtcm[3]`
Definition at line 1341 of file rtklib.h.

12.345.2.26 rtk

`rtk_t rtksvr_t::rtk`
Definition at line 1331 of file rtklib.h.

12.345.2.27 sbsmsg

`sbsmsg_t rtksvr_t::sbsmsg[MAXSBSMSG]`
Definition at line 1346 of file rtklib.h.

12.345.2.28 sbuf

`unsigned char* rtksvr_t::sbuf[2]`
Definition at line 1336 of file rtklib.h.

12.345.2.29 solbuf

`sol_t rtksvr_t::solbuf[MAXSOLBUF]`
Definition at line 1338 of file rtklib.h.

12.345.2.30 solopt

`solopt_t rtksvr_t::solopt[2]`
Definition at line 1327 of file rtklib.h.

12.345.2.31 state

`int rtksvr_t::state`
Definition at line 1320 of file rtklib.h.

12.345.2.32 stream

`stream_t rtksvr_t::stream[8]`
Definition at line 1347 of file rtklib.h.

12.345.2.33 thread

`pthread_t rtksvr_t::thread`
Definition at line 1350 of file rtklib.h.

12.345.2.34 tick

`unsigned int rtksvr_t::tick`
Definition at line 1349 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.346 Rtl_Tcp_Dongle_Info Class Reference

This class represents the dongle information which is sent by rtl_tcp.
`#include <rtl_tcp_dongle_info.h>`

Public Types

- enum {
  TUNER_UNKNOWN = 0 , **TUNER_E4000** , **TUNER_FC0012** , **TUNER_FC0013** ,
  TUNER_FC2580 , **TUNER_R820T** , **TUNER_R828D** }

Public Member Functions

- boost::system::error_code **read** (boost::asio::ip::tcp::socket &socket)
- bool **is_valid** () const
- const char * **get_type_name** () const
- double **clip_gain** (int gain) const
- uint32_t get_tuner_type () const
- uint32_t get_tuner_gain_count () const

12.346.1 Detailed Description

This class represents the dongle information which is sent by rtl_tcp.
Definition at line 35 of file rtl_tcp_dongle_info.h.

12.346.2 Member Enumeration Documentation

12.346.2.1 anonymous enum

```
anonymous enum
```
Definition at line 38 of file rtl_tcp_dongle_info.h.

12.346.3 Member Function Documentation

12.346.3.1 get_tuner_gain_count()

```
uint32_t Rtl_Tcp_Dongle_Info::get_tuner_gain_count () const  [inline]
```
Definition at line 64 of file rtl_tcp_dongle_info.h.

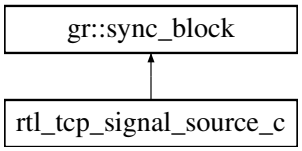
12.346.3.2 get_tuner_type()

```
uint32_t Rtl_Tcp_Dongle_Info::get_tuner_type () const  [inline]
```
Definition at line 59 of file rtl_tcp_dongle_info.h.
The documentation for this class was generated from the following file:

- rtl_tcp_dongle_info.h

12.347 rtl_tcp_signal_source_c Class Reference

This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types.
`#include <rtl_tcp_signal_source_c.h>`
Inheritance diagram for rtl_tcp_signal_source_c:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)
- void **set_frequency** (int frequency)
- void **set_sample_rate** (int sample_rate)
- void **set_agc_mode** (bool agc)
- void **set_gain** (int gain)
- void **set_if_gain** (int gain)

Friends

- rtl_tcp_signal_source_c_sptr **rtl_tcp_make_signal_source_c** (const std::string &address, int16_t port, bool flip_iq=false)

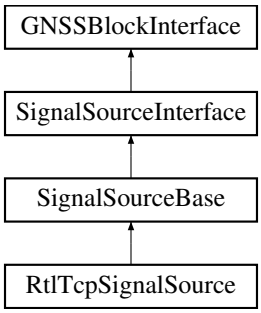
12.347.1 Detailed Description

This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types.
Definition at line 65 of file rtl_tcp_signal_source_c.h.
The documentation for this class was generated from the following file:

- rtl_tcp_signal_source_c.h

12.348 RtlTcpSignalSource Class Reference

This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see https://osmocom.org/projects/rtl-sdr/wiki).
`#include <rtl_tcp_signal_source.h>`
Inheritance diagram for RtlTcpSignalSource:



Public Member Functions

- **RtlTcpSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.348.1 Detailed Description

This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see https://osmocom.↩ org/projects/rtl-sdr/wiki).
Definition at line 45 of file rtl_tcp_signal_source.h.

12.348.2 Member Function Documentation

12.348.2.1 connect()

```
void RtlTcpSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.348.2.2 disconnect()

```
void RtlTcpSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.348.2.3 get_left_block()

```
gr::basic_block_sptr RtlTcpSignalSource::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.348.2.4 get_right_block()

```
gr::basic_block_sptr RtlTcpSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.348.2.5 item_size()

```
size_t RtlTcpSignalSource::item_size ()  [inline], [override], [virtual]
```

Implements GNSSBlockInterface.
Definition at line 56 of file rtl_tcp_signal_source.h.
The documentation for this class was generated from the following file:

- rtl_tcp_signal_source.h

12.349 SatelliteVisibility Class Reference

Opt-in acquisition search classification: visible, excluded, or unknown. Healthy satellites above the search mask are preferred; unknowns are searched less often. Banned PRNs are omitted, and tracking can override exclusion. Tick() refreshes on reference, fix, navigation-data, position, interval, or expiry changes. Expired data returns satellites to the unknown pool. Disabled by default (GNSS-SDR.enable_visibility_aware_search).
`#include <satellite_visibility.h>`

Public Member Functions

- **SatelliteVisibility** (const std::shared_ptr< ConfigurationInterface > &configuration)
- bool enabled () const
- uint32_t search_ratio () const
- void SetCommandReference (time_t utc_time, const std::array< float, 3 > &LLH, const Monitor_Pvt &current_fix, double receiver_time_s)

  Supplies a telecommand position and UTC epoch, anchored to the current sample clock. Forces the next Tick() to recompute and overrides the current fix/configured reference until a different PVT epoch arrives.
- bool Tick (const std::shared_ptr< PvtInterface > &pvt_ptr, const Monitor_Pvt &fix_status, double receiver_time_s=0.0)

  Refresh visibility using telecommand, fix, then AGNSS reference priority.
- bool **IsVisible** (const Gnss_Satellite &sat) const
- bool **IsExcluded** (const Gnss_Satellite &sat) const
- bool **IsSearchVisible** (const Gnss_Satellite &sat) const
- bool **IsSearchExcluded** (const Gnss_Satellite &sat) const
- bool **PredictedDopplerHz** (const std::shared_ptr< PvtInterface > &pvt_ptr, const Monitor_Pvt &fix_status, double receiver_time_s, const Gnss_Satellite &sat, const std::string &signal, double &doppler_hz) const
- double **PredictedDopplerUncertaintyHz** (const Monitor_Pvt &fix_status, const std::string &signal) const

12.349.1 Detailed Description

Opt-in acquisition search classification: visible, excluded, or unknown. Healthy satellites above the search mask are preferred; unknowns are searched less often. Banned PRNs are omitted, and tracking can override exclusion. Tick() refreshes on reference, fix, navigation-data, position, interval, or expiry changes. Expired data returns satellites to the unknown pool. Disabled by default (GNSS-SDR.enable_visibility_aware_search).
Definition at line 82 of file satellite_visibility.h.

12.349.2 Member Function Documentation

12.349.2.1 enabled()

```
bool SatelliteVisibility::enabled () const  [inline]
```

Definition at line 87 of file satellite_visibility.h.

12.349.2.2 search_ratio()

```
uint32_t SatelliteVisibility::search_ratio () const  [inline]
```

Definition at line 88 of file satellite_visibility.h.

12.349.2.3 SetCommandReference()

```
void SatelliteVisibility::SetCommandReference (
            time_t utc_time,
            const std::array< float, 3 > & LLH,
            const Monitor_Pvt & current_fix,
            double receiver_time_s)
```

Supplies a telecommand position and UTC epoch, anchored to the current sample clock. Forces the next Tick() to recompute and overrides the current fix/configured reference until a different PVT epoch arrives.

12.349.2.4 Tick()

```
bool SatelliteVisibility::Tick (
            const std::shared_ptr< PvtInterface > & pvt_ptr,
            const Monitor_Pvt & fix_status,
            double receiver_time_s = 0.0)
```

Refresh visibility using telecommand, fix, then AGNSS reference priority.

Parameters

receiver_time_s	elapsed sample time; advances epochs between fixes.

Returns

true when classification changes require rebuilding search queues.

The documentation for this class was generated from the following file:

- satellite_visibility.h

12.350 Sbas_Ephemeris Class Reference

This class stores SBAS SV ephemeris data.
`#include <sbas_ephemeris.h>`

Public Member Functions

- void **print** (std::ostream &out)

Public Attributes

- int i_prn {}
  PRN number.
- int i_t0 {}
  Reference epoch time (GPST).
- double d_tof {}
  Time of message frame (GPST).
- int i_sv_ura {}
  SV accuracy (URA index), not standardized.
- bool b_sv_do_not_use {}
  Health status (false:do not use / true:usable).
- double d_pos [3] {}
  Satellite position (m) (ECEF).
- double d_vel [3] {}
  Satellite velocity (m/s) (ECEF).
- double d_acc [3] {}
  Satellite acceleration (m/s^2) (ECEF).
- double d_af0 {}
  Satellite clock-offset (s).
- double d_af1 {}
  Satellite drift (s/s).

12.350.1 Detailed Description

This class stores SBAS SV ephemeris data.
Definition at line 33 of file sbas_ephemeris.h.

12.350.2 Member Data Documentation

12.350.2.1 b_sv_do_not_use

`bool Sbas_Ephemeris::b_sv_do_not_use {}`
Health status (false:do not use / true:usable).
Definition at line 42 of file sbas_ephemeris.h.

12.350.2.2 d_acc

```
double Sbas_Ephemeris::d_acc[3] {}
```
Satellite acceleration (m/s^2) (ECEF).
Definition at line 45 of file sbas_ephemeris.h.

12.350.2.3 d_af0

```
double Sbas_Ephemeris::d_af0 {}
```
Satellite clock-offset (s).
Definition at line 46 of file sbas_ephemeris.h.

12.350.2.4 d_af1

```
double Sbas_Ephemeris::d_af1 {}
```
Satellite drift (s/s).
Definition at line 47 of file sbas_ephemeris.h.

12.350.2.5 d_pos

```
double Sbas_Ephemeris::d_pos[3] {}
```
Satellite position (m) (ECEF).
Definition at line 43 of file sbas_ephemeris.h.

12.350.2.6 d_tof

```
double Sbas_Ephemeris::d_tof {}
```
Time of message frame (GPST).
Definition at line 40 of file sbas_ephemeris.h.

12.350.2.7 d_vel

```
double Sbas_Ephemeris::d_vel[3] {}
```
Satellite velocity (m/s) (ECEF).
Definition at line 44 of file sbas_ephemeris.h.

12.350.2.8 i_prn

```
int Sbas_Ephemeris::i_prn {}
```
PRN number.
Definition at line 38 of file sbas_ephemeris.h.

12.350.2.9 i_sv_ura

```
int Sbas_Ephemeris::i_sv_ura {}
```
SV accuracy (URA index), not standardized.
Definition at line 41 of file sbas_ephemeris.h.

12.350.2.10 i_t0

```
int Sbas_Ephemeris::i_t0 {}
```
Reference epoch time (GPST).
Definition at line 39 of file sbas_ephemeris.h.
The documentation for this class was generated from the following file:

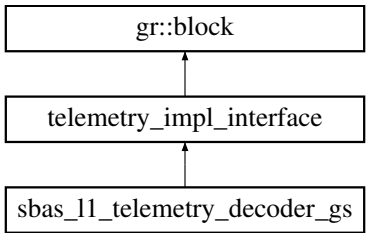
- sbas_ephemeris.h

12.351 sbas_l1_telemetry_decoder_gs Class Reference

This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229.

`#include <sbas_l1_telemetry_decoder_gs.h>`

Inheritance diagram for sbas_l1_telemetry_decoder_gs:



Public Member Functions

- void set_satellite (const Gnss_Satellite &satellite) override

  Set satellite PRN.
- void set_channel (int32_t channel) override

  Set receiver's channel Clears all internal decoding state (buffers, aligners, Viterbi decoders) and closes the dump file, so that a channel reassigned to a different PRN never mixes messages from the previous satellite into its file.
- void reset () override
- int general_work (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

  This is where all signal processing takes place.

Public Member Functions inherited from telemetry_impl_interface

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_channel (int channel)=0

Static Public Member Functions

- static std::vector< double > consume_decoded_timestamps (std::deque< double > &pending_stamps, const std::vector< double > &input_stamps, int32_t decoded_bits)

  Exposed for unit testing: appends timestamps entering a continuous decoder and returns the timestamps corresponding to the decoded bits.

Friends

- sbas_l1_telemetry_decoder_gs_sptr **sbas_l1_make_telemetry_decoder_gs** (bool dump, std::string dump_filename=std::string())

Additional Inherited Members

Protected Member Functions inherited from telemetry_impl_interface

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.351.1 Detailed Description

This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229.
Definition at line 70 of file sbas_l1_telemetry_decoder_gs.h.

12.351.2 Member Function Documentation

12.351.2.1 consume_decoded_timestamps()

```
std::vector< double > sbas_l1_telemetry_decoder_gs::consume_decoded_timestamps (
            std::deque< double > & pending_stamps,
            const std::vector< double > & input_stamps,
            int32_t decoded_bits)  [static]
```

Exposed for unit testing: appends timestamps entering a continuous decoder and returns the timestamps corresponding to the decoded bits.

12.351.2.2 general_work()

```
int sbas_l1_telemetry_decoder_gs::general_work (
            int noutput_items,
            gr_vector_int & ninput_items,
            gr_vector_const_void_star & input_items,
            gr_vector_void_star & output_items)  [override]
```

This is where all signal processing takes place.

12.351.2.3 reset()

```
void sbas_l1_telemetry_decoder_gs::reset ()  [override], [virtual]
```

Implements telemetry_impl_interface.

12.351.2.4 set_channel()

```
void sbas_l1_telemetry_decoder_gs::set_channel (
            int32_t channel)  [override]
```

Set receiver's channel Clears all internal decoding state (buffers, aligners, Viterbi decoders) and closes the dump file, so that a channel reassigned to a different PRN never mixes messages from the previous satellite into its file.

12.351.2.5 set_satellite()

```
void sbas_l1_telemetry_decoder_gs::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Set satellite PRN.
Implements telemetry_impl_interface.
The documentation for this class was generated from the following file:

- sbas_l1_telemetry_decoder_gs.h

12.352 Sbas_Raw_Message Class Reference

A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp.
`#include <sbas_raw_message.h>`

Public Member Functions

- Sbas_Raw_Message (double sample_stamp_s, uint32_t prn, const std::vector< uint8_t > &frame)
- double sample_stamp_s () const
- uint32_t prn () const
- uint32_t message_type () const
- const std::array< uint8_t, SBAS_L1_MSG_LENGTH_BYTES > & frame () const

12.352.1 Detailed Description

A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp.
The frame is stored MSB first as 32 bytes. Its six least-significant padding bits are zero. The timestamp identifies the first bit of the preamble in the receiver sample-clock time base.
Definition at line 39 of file sbas_raw_message.h.

12.352.2 Constructor & Destructor Documentation

12.352.2.1 Sbas_Raw_Message()

```
Sbas_Raw_Message::Sbas_Raw_Message (
            double sample_stamp_s,
            uint32_t prn,
            const std::vector< uint8_t > & frame)  [inline]
```

Definition at line 44 of file sbas_raw_message.h.

12.352.3 Member Function Documentation

12.352.3.1 frame()

```
const std::array< uint8_t, SBAS_L1_MSG_LENGTH_BYTES > & Sbas_Raw_Message::frame () const
[inline]
```

Definition at line 56 of file sbas_raw_message.h.

12.352.3.2 message_type()

```
uint32_t Sbas_Raw_Message::message_type () const  [inline]
```

Definition at line 55 of file sbas_raw_message.h.

12.352.3.3 prn()

```
uint32_t Sbas_Raw_Message::prn () const  [inline]
```

Definition at line 54 of file sbas_raw_message.h.

12.352.3.4 sample_stamp_s()

```
double Sbas_Raw_Message::sample_stamp_s () const  [inline]
```

Definition at line 53 of file sbas_raw_message.h.
The documentation for this class was generated from the following file:

- sbas_raw_message.h

12.353 Sbas_Rtklib_Corrections Class Reference

Maintains RTKLIB SBAS state from timestamped raw L1 messages.
`#include <sbas_rtklib_corrections.h>`

Public Member Functions

- **Sbas_Rtklib_Corrections** (int32_t selected_prn=0)
- **Sbas_Rtklib_Corrections** (const Sbas_Rtklib_Corrections &)=delete
- Sbas_Rtklib_Corrections & **operator=** (const Sbas_Rtklib_Corrections &)=delete
- void **push** (const Sbas_Raw_Message &message)
- void **update** (int32_t gps_week, double gps_tow_s, double receiver_sample_stamp_s)
- void **copy_to** (nav_t &navigation_data)
- int32_t active_prn () const
- size_t pending_messages () const
- const sbssat_t & satellite_corrections () const
- const sbsion_t * ionosphere_corrections () const

12.353.1 Detailed Description

Maintains RTKLIB SBAS state from timestamped raw L1 messages.
Raw messages use the receiver sample-clock time base. update() relates that clock to GPST at an observation epoch before passing messages to RTKLIB.
Definition at line 38 of file sbas_rtklib_corrections.h.

12.353.2 Member Function Documentation

12.353.2.1 active_prn()

```
int32_t Sbas_Rtklib_Corrections::active_prn () const  [inline]
```
Definition at line 50 of file sbas_rtklib_corrections.h.

12.353.2.2 ionosphere_corrections()

```
const sbsion_t * Sbas_Rtklib_Corrections::ionosphere_corrections () const  [inline]
```
Definition at line 53 of file sbas_rtklib_corrections.h.

12.353.2.3 pending_messages()

```
size_t Sbas_Rtklib_Corrections::pending_messages () const  [inline]
```
Definition at line 51 of file sbas_rtklib_corrections.h.

12.353.2.4 satellite_corrections()

```
const sbssat_t & Sbas_Rtklib_Corrections::satellite_corrections () const  [inline]
```
Definition at line 52 of file sbas_rtklib_corrections.h.
The documentation for this class was generated from the following file:

- sbas_rtklib_corrections.h

12.354 SbasL1DumpSession Class Reference

Public Member Functions

- **SbasL1DumpSession** (const SbasL1DumpSession &)=delete
- SbasL1DumpSession & **operator=** (const SbasL1DumpSession &)=delete
- bool open (std::ofstream &stream, const std::string &filename)

  Opens filename with truncation on first use and append on later uses. Returns true on first use; stream state reports whether opening succeeded.

12.354.1 Detailed Description

Definition at line 42 of file sbas_l1_telemetry_decoder_gs.h.

12.354.2 Member Function Documentation

12.354.2.1 open()

```
bool SbasL1DumpSession::open (
            std::ofstream & stream,
            const std::string & filename)
```
Opens filename with truncation on first use and append on later uses. Returns true on first use; stream state reports whether opening succeeded.
The documentation for this class was generated from the following file:

- sbas_l1_telemetry_decoder_gs.h

12.355 sbs_t Struct Reference

Public Attributes

- int n
- int nmax
- sbsmsg_t * msgs

12.355.1 Detailed Description

Definition at line 609 of file rtklib.h.

12.355.2 Member Data Documentation

12.355.2.1 msgs

```
sbsmsg_t* sbs_t::msgs
```

Definition at line 612 of file rtklib.h.

12.355.2.2 n

```
int sbs_t::n
```

Definition at line 611 of file rtklib.h.

12.355.2.3 nmax

```
int sbs_t::nmax
```

Definition at line 611 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.356 sbsfcorr_t Struct Reference

Public Attributes

- gtime_t t0
- double prc
- double rrc
- double dt
- int iodf
- short udre
- short ai

12.356.1 Detailed Description

Definition at line 616 of file rtklib.h.

12.356.2 Member Data Documentation

12.356.2.1 ai

```
short sbsfcorr_t::ai
```

Definition at line 624 of file rtklib.h.

12.356.2.2 dt

```
double sbsfcorr_t::dt
```

Definition at line 621 of file rtklib.h.

12.356.2.3 iodf

```
int sbsfcorr_t::iodf
```
Definition at line 622 of file rtklib.h.

12.356.2.4 prc

```
double sbsfcorr_t::prc
```
Definition at line 619 of file rtklib.h.

12.356.2.5 rrc

```
double sbsfcorr_t::rrc
```
Definition at line 620 of file rtklib.h.

12.356.2.6 t0

```
gtime_t sbsfcorr_t::t0
```
Definition at line 618 of file rtklib.h.

12.356.2.7 udre

```
short sbsfcorr_t::udre
```
Definition at line 623 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.357 sbsigp_t Struct Reference

Public Attributes

- gtime_t t0
- short lat
- short lon
- short give
- float delay

12.357.1 Detailed Description

Definition at line 655 of file rtklib.h.

12.357.2 Member Data Documentation

12.357.2.1 delay

```
float sbsigp_t::delay
```
Definition at line 660 of file rtklib.h.

12.357.2.2 give

```
short sbsigp_t::give
```
Definition at line 659 of file rtklib.h.

12.357.2.3 lat

```
short sbsigp_t::lat
```
Definition at line 658 of file rtklib.h.

12.357.2.4 lon

```
short sbsigp_t::lon
```
Definition at line 658 of file rtklib.h.

12.357.2.5 t0

```
gtime_t sbsigp_t::t0
```
Definition at line 657 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.358 sbsigpband_t Struct Reference

Public Attributes

- short x
- const short * y
- unsigned char bits
- unsigned char bite

12.358.1 Detailed Description

Definition at line 664 of file rtklib.h.

12.358.2 Member Data Documentation

12.358.2.1 bite

```
unsigned char sbsigpband_t::bite
```
Definition at line 669 of file rtklib.h.

12.358.2.2 bits

```
unsigned char sbsigpband_t::bits
```
Definition at line 668 of file rtklib.h.

12.358.2.3 x

```
short sbsigpband_t::x
```
Definition at line 666 of file rtklib.h.

12.358.2.4 y

```
const short* sbsigpband_t::y
```
Definition at line 667 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.359 sbsion_t Struct Reference

Public Attributes

- int iodi
- int nigp
- sbsigp_t igp [MAXNIGP]

12.359.1 Detailed Description

Definition at line 673 of file rtklib.h.

12.359.2 Member Data Documentation

12.359.2.1 igp

`sbsigp_t sbsion_t::igp[MAXNIGP]`
Definition at line 677 of file rtklib.h.

12.359.2.2 iodi

`int sbsion_t::iodi`
Definition at line 675 of file rtklib.h.

12.359.2.3 nigp

`int sbsion_t::nigp`
Definition at line 676 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.360 sbslcorr_t Struct Reference

Public Attributes

- gtime_t t0
- int iode
- double dpos [3]
- double dvel [3]
- double daf0
- double daf1

12.360.1 Detailed Description

Definition at line 628 of file rtklib.h.

12.360.2 Member Data Documentation

12.360.2.1 daf0

`double sbslcorr_t::daf0`
Definition at line 634 of file rtklib.h.

12.360.2.2 daf1

`double sbslcorr_t::daf1`
Definition at line 634 of file rtklib.h.

12.360.2.3 dpos

`double sbslcorr_t::dpos[3]`
Definition at line 632 of file rtklib.h.

12.360.2.4 dvel

`double sbslcorr_t::dvel[3]`
Definition at line 633 of file rtklib.h.

12.360.2.5 iode

`int sbslcorr_t::iode`
Definition at line 631 of file rtklib.h.

12.360.2.6 t0

`gtime_t sbslcorr_t::t0`
Definition at line 630 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.361 sbsmsg_t Struct Reference

Public Attributes

- int week
- int tow
- int prn
- unsigned char msg [29]

12.361.1 Detailed Description

Definition at line 601 of file rtklib.h.

12.361.2 Member Data Documentation

12.361.2.1 msg

`unsigned char sbsmsg_t::msg[29]`
Definition at line 605 of file rtklib.h.

12.361.2.2 prn

`int sbsmsg_t::prn`
Definition at line 604 of file rtklib.h.

12.361.2.3 tow

`int sbsmsg_t::tow`
Definition at line 603 of file rtklib.h.

12.361.2.4 week

`int sbsmsg_t::week`
Definition at line 603 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.362 sbssat_t Struct Reference

Public Attributes

- int iodp
- int nsat
- int tlat
- sbssatp_t sat [MAXSAT]

12.362.1 Detailed Description

Definition at line 646 of file rtklib.h.

12.362.2 Member Data Documentation

12.362.2.1 iodp

```
int sbssat_t::iodp
```

Definition at line 648 of file rtklib.h.

12.362.2.2 nsat

```
int sbssat_t::nsat
```

Definition at line 649 of file rtklib.h.

12.362.2.3 sat

```
sbssatp_t sbssat_t::sat[MAXSAT]
```

Definition at line 651 of file rtklib.h.

12.362.2.4 tlat

```
int sbssat_t::tlat
```

Definition at line 650 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.363 sbssatp_t Struct Reference

Public Attributes

- int sat
- sbsfcorr_t fcorr
- sbslcorr_t lcorr

12.363.1 Detailed Description

Definition at line 638 of file rtklib.h.

12.363.2 Member Data Documentation

12.363.2.1 fcorr

```
sbsfcorr_t sbssatp_t::fcorr
```

Definition at line 641 of file rtklib.h.

12.363.2.2 lcorr

```
sbslcorr_t sbssatp_t::lcorr
```

Definition at line 642 of file rtklib.h.

12.363.2.3 sat

```
int sbssatp_t::sat
```

Definition at line 640 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.364 sdr_gnss_packet_t Struct Reference

Classes

- struct [struct].sats

Public Attributes

- uint8_t nsvfix
- uint8_t gps_satellites
- uint8_t galileo_satellites
- uint32_t microseconds
- double latitude
- double longitude
- double height
- float velocity [3]
- struct **sats** [16]
- uint32_t reserved
- uint16_t status

12.364.1 Detailed Description

Definition at line 39 of file an_packet_printer.h.

12.364.2 Member Data Documentation

12.364.2.1 galileo_satellites

```
uint8_t sdr_gnss_packet_t::galileo_satellites
```

Definition at line 43 of file an_packet_printer.h.

12.364.2.2 gps_satellites

```
uint8_t sdr_gnss_packet_t::gps_satellites
```

Definition at line 42 of file an_packet_printer.h.

12.364.2.3 height

```
double sdr_gnss_packet_t::height
```

Definition at line 47 of file an_packet_printer.h.

12.364.2.4 latitude

```
double sdr_gnss_packet_t::latitude
```

Definition at line 45 of file an_packet_printer.h.

12.364.2.5 longitude

```
double sdr_gnss_packet_t::longitude
```

Definition at line 46 of file an_packet_printer.h.

12.364.2.6 microseconds

```
uint32_t sdr_gnss_packet_t::microseconds
```

Definition at line 44 of file an_packet_printer.h.

12.364.2.7 nsvfix

```
uint8_t sdr_gnss_packet_t::nsvfix
```

Definition at line 41 of file an_packet_printer.h.

12.364.2.8 reserved

```
uint32_t sdr_gnss_packet_t::reserved
```
Definition at line 57 of file an_packet_printer.h.

12.364.2.9 status

```
uint16_t sdr_gnss_packet_t::status
```
Definition at line 58 of file an_packet_printer.h.

12.364.2.10 velocity

```
float sdr_gnss_packet_t::velocity[3]
```
Definition at line 48 of file an_packet_printer.h.
The documentation for this struct was generated from the following file:

- an_packet_printer.h

12.365 Secure_String Class Reference

Value type for credential-bearing configuration fields.
`#include <secure_string.h>`

Public Member Functions

- Secure_String (std::string value)
- Secure_String (const char *value)
- **Secure_String** (const Secure_String &other)=default
- Secure_String (Secure_String &&other) noexcept
- Secure_String & operator= (const Secure_String &other)
- Secure_String & operator= (Secure_String &&other) noexcept
- Secure_String & operator= (std::string value)
- void clear () noexcept
- const std::string & value () const noexcept
- bool empty () const noexcept

12.365.1 Detailed Description

Value type for credential-bearing configuration fields.
Wipes its storage with volatile stores before every release - destruction, assignment, clear() - so every copy of a configuration struct that carries one scrubs itself, and a new holder of such a copy needs no manual scrub site. Assign-once usage is assumed: in-place growth of the underlying std::string is not protected.
Definition at line 39 of file secure_string.h.

12.365.2 Constructor & Destructor Documentation

12.365.2.1 Secure_String() [1/3]

```
Secure_String::Secure_String (
            std::string value)  [inline], [explicit]
```
Definition at line 44 of file secure_string.h.

12.365.2.2 Secure_String() [2/3]

```
Secure_String::Secure_String (
            const char * value)  [inline], [explicit]
```
Definition at line 51 of file secure_string.h.

12.365.2.3 Secure_String() [3/3]

```
Secure_String::Secure_String (
            Secure_String && other)  [inline], [noexcept]
```

Definition at line 55 of file secure_string.h.

12.365.2.4 ~Secure_String()

```
Secure_String::~Secure_String ()  [inline]
```

Definition at line 90 of file secure_string.h.

12.365.3 Member Function Documentation

12.365.3.1 clear()

```
void Secure_String::clear ()  [inline], [noexcept]
```

Definition at line 95 of file secure_string.h.

12.365.3.2 empty()

```
bool Secure_String::empty () const  [inline], [noexcept]
```

Definition at line 105 of file secure_string.h.

12.365.3.3 operator=() [1/3]

```
Secure_String & Secure_String::operator= (
            const Secure_String & other)  [inline]
```

Definition at line 60 of file secure_string.h.

12.365.3.4 operator=() [2/3]

```
Secure_String & Secure_String::operator= (
            Secure_String && other)  [inline], [noexcept]
```

Definition at line 70 of file secure_string.h.

12.365.3.5 operator=() [3/3]

```
Secure_String & Secure_String::operator= (
            std::string value)  [inline]
```

Definition at line 81 of file secure_string.h.

12.365.3.6 value()

```
const std::string & Secure_String::value () const  [inline], [noexcept]
```

Definition at line 100 of file secure_string.h.
The documentation for this class was generated from the following file:

- secure_string.h

12.366 IONGSMSFileSource::SegmentDescriptor Struct Reference

Public Attributes

- fs::path data_filepath
- const GnssMetadata::Block * block = nullptr
- std::size_t block_start_offset = 0
- bool block_extends_to_eof = false

12.366.1 Detailed Description

Definition at line 42 of file ion_gsms.h.

12.366.2 Member Data Documentation

12.366.2.1 block

```
const GnssMetadata::Block* IONGSMSFileSource::SegmentDescriptor::block = nullptr
```
Definition at line 45 of file ion_gsms.h.

12.366.2.2 block_extends_to_eof

```
bool IONGSMSFileSource::SegmentDescriptor::block_extends_to_eof = false
```
Definition at line 47 of file ion_gsms.h.

12.366.2.3 block_start_offset

```
std::size_t IONGSMSFileSource::SegmentDescriptor::block_start_offset = 0
```
Definition at line 46 of file ion_gsms.h.

12.366.2.4 data_filepath

```
fs::path IONGSMSFileSource::SegmentDescriptor::data_filepath
```
Definition at line 44 of file ion_gsms.h.
The documentation for this struct was generated from the following file:

- ion_gsms.h

12.367 SensorDataAggregator Class Reference

Public Member Functions

- **SensorDataAggregator** (const SensorDataSourceConfiguration &configuration, const std::vector< SensorIdentifier::value_type > &required_sensors)
- void **update** (const std::vector< gr::tag_t > &tags)
- const std::vector< SensorDataSample< float > > & **get_f32** (SensorIdentifier::value_type sensor_id) const
- SensorDataSample< float > **get_last_f32** (SensorIdentifier::value_type sensor_id) const
- SensorDataSample< float > **get_average_f32** (SensorIdentifier::value_type sensor_id) const
- const std::vector< SensorDataSample< double > > & **get_f64** (SensorIdentifier::value_type sensor_id) const
- SensorDataSample< double > **get_last_f64** (SensorIdentifier::value_type sensor_id) const
- SensorDataSample< double > **get_average_f64** (SensorIdentifier::value_type sensor_id) const

Public Attributes

- const pmt::pmt_t SENSOR_DATA_TAG = pmt::mp("sensor_data")

12.367.1 Detailed Description

Definition at line 45 of file sensor_data_aggregator.h.

12.367.2 Member Data Documentation

12.367.2.1 SENSOR_DATA_TAG

```
const pmt::pmt_t SensorDataAggregator::SENSOR_DATA_TAG = pmt::mp("sensor_data")
```
Definition at line 48 of file sensor_data_aggregator.h.
The documentation for this class was generated from the following file:

- sensor_data_aggregator.h

12.368 SensorDataConfiguration Struct Reference

Public Attributes

- uint64_t id
- uint64_t file_id
- uint64_t offset
- SensorIdentifier::value_type identifier
- SensorDataType::value_type type
- pmt::pmt_t tag_key

12.368.1 Detailed Description

Definition at line 46 of file sensor_data_source_configuration.h.

12.368.2 Member Data Documentation

12.368.2.1 file_id

```
uint64_t SensorDataConfiguration::file_id
```

Definition at line 49 of file sensor_data_source_configuration.h.

12.368.2.2 id

```
uint64_t SensorDataConfiguration::id
```

Definition at line 48 of file sensor_data_source_configuration.h.

12.368.2.3 identifier

```
SensorIdentifier::value_type SensorDataConfiguration::identifier
```

Definition at line 51 of file sensor_data_source_configuration.h.

12.368.2.4 offset

```
uint64_t SensorDataConfiguration::offset
```

Definition at line 50 of file sensor_data_source_configuration.h.

12.368.2.5 tag_key

```
pmt::pmt_t SensorDataConfiguration::tag_key
```

Definition at line 53 of file sensor_data_source_configuration.h.

12.368.2.6 type

```
SensorDataType::value_type SensorDataConfiguration::type
```

Definition at line 52 of file sensor_data_source_configuration.h.
The documentation for this struct was generated from the following file:

- sensor_data_source_configuration.h

12.369 SensorDataFile Class Reference

Public Types

- using sptr = gnss_shared_ptr<SensorDataFile>
- using id_type = std::size_t

Public Member Functions

- **SensorDataFile** (const std::string &path, const std::size_t &sample_delay, const std::size_t &sample_period, const std::size_t &offset_in_file, const std::size_t &item_size, const bool &repeat)
- void **reset** ()
- bool **read_until_sample** (std::size_t end_sample, std::size_t &sample_stamp, std::vector< uint8_t > &buffer)
- std::size_t **get_chunks_read** () const

12.369.1 Detailed Description

Definition at line 35 of file sensor_data_file.h.

12.369.2 Member Typedef Documentation

12.369.2.1 id_type

```
using SensorDataFile::id_type = std::size_t
```

Definition at line 39 of file sensor_data_file.h.

12.369.2.2 sptr

```
using SensorDataFile::sptr = gnss_shared_ptr<SensorDataFile>
```

Definition at line 38 of file sensor_data_file.h.
The documentation for this class was generated from the following file:

- sensor_data_file.h

12.370 SensorDataFileConfiguration Struct Reference

Public Attributes

- uint64_t id
- uint64_t chunk_size
- uint64_t file_offset
- uint64_t sample_offset
- uint64_t sample_period
- std::string filename
- bool repeat

12.370.1 Detailed Description

Definition at line 34 of file sensor_data_source_configuration.h.

12.370.2 Member Data Documentation

12.370.2.1 chunk_size

```
uint64_t SensorDataFileConfiguration::chunk_size
```

Definition at line 37 of file sensor_data_source_configuration.h.

12.370.2.2 file_offset

```
uint64_t SensorDataFileConfiguration::file_offset
```

Definition at line 38 of file sensor_data_source_configuration.h.

12.370.2.3 filename

```
std::string SensorDataFileConfiguration::filename
```

Definition at line 41 of file sensor_data_source_configuration.h.

12.370.2.4 id

```
uint64_t SensorDataFileConfiguration::id
```
Definition at line 36 of file sensor_data_source_configuration.h.

12.370.2.5 repeat

```
bool SensorDataFileConfiguration::repeat
```
Definition at line 42 of file sensor_data_source_configuration.h.

12.370.2.6 sample_offset

```
uint64_t SensorDataFileConfiguration::sample_offset
```
Definition at line 39 of file sensor_data_source_configuration.h.

12.370.2.7 sample_period

```
uint64_t SensorDataFileConfiguration::sample_period
```
Definition at line 40 of file sensor_data_source_configuration.h.
The documentation for this struct was generated from the following file:

- sensor_data_source_configuration.h

12.371 SensorDataSample< DataType > Struct Template Reference

Public Member Functions

- constexpr SensorDataSample (uint64_t t, DataType v)

Public Attributes

- uint64_t timestamp
- DataType value

12.371.1 Detailed Description

template<typename DataType>
struct SensorDataSample< DataType >

Definition at line 35 of file sensor_data_aggregator.h.

12.371.2 Constructor & Destructor Documentation

12.371.2.1 SensorDataSample()

```
template<typename DataType>
SensorDataSample< DataType >::SensorDataSample (
            uint64_t t,
            DataType v)  [inline], [constexpr]
```
Definition at line 40 of file sensor_data_aggregator.h.

12.371.3 Member Data Documentation

12.371.3.1 timestamp

```
template<typename DataType>
uint64_t SensorDataSample< DataType >::timestamp
```
Definition at line 37 of file sensor_data_aggregator.h.

12.371.3.2 value

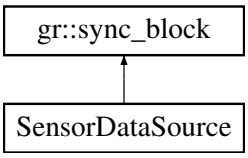
```
template<typename DataType>
DataType SensorDataSample< DataType >::value
```

Definition at line 38 of file sensor_data_aggregator.h.
The documentation for this struct was generated from the following file:

- sensor_data_aggregator.h

12.372 SensorDataSource Class Reference

Inheritance diagram for SensorDataSource:



Public Types

- using sptr = gnss_shared_ptr<SensorDataSource>

Public Member Functions

- **SensorDataSource** (const SensorDataSourceConfiguration &configuration, const gr::io_signature::sptr &io_signature)
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override

12.372.1 Detailed Description

Definition at line 33 of file sensor_data_source.h.

12.372.2 Member Typedef Documentation

12.372.2.1 sptr

```
using SensorDataSource::sptr = gnss_shared_ptr<SensorDataSource>
```

Definition at line 36 of file sensor_data_source.h.
The documentation for this class was generated from the following file:

- sensor_data_source.h

12.373 SensorDataSourceConfiguration Class Reference

Public Member Functions

- **SensorDataSourceConfiguration** (const ConfigurationInterface *configuration)
- bool **validate** () const
- bool **is_enabled** () const
- bool **is_sensor_provided** (SensorIdentifier::value_type sensor_id) const
- const std::unordered_map< uint64_t, SensorDataFileConfiguration > & **files** () const
- const std::vector< SensorDataConfiguration > & **sensors** () const
- void **set_items_per_sample** (uint64_t items_per_sample)
- uint64_t **get_items_per_sample** () const

12.373.1 Detailed Description

Definition at line 57 of file sensor_data_source_configuration.h.
The documentation for this class was generated from the following file:

- sensor_data_source_configuration.h

12.374 SensorDataType Struct Reference

Public Types

- enum **value_type** {
  UINT64 , **F32** , **F64** , **I32** ,
  I64 }

Static Public Member Functions

- static value_type **from_string** (const std::string &s)
- static std::string **to_string** (const value_type &v)
- static uint64_t **get_size** (const value_type &v)
- static pmt::pmt_t **make_value** (const value_type &v, void ∗value)

12.374.1 Detailed Description

Definition at line 29 of file sensor_data_type.h.

12.374.2 Member Enumeration Documentation

12.374.2.1 value_type

`enum SensorDataType::value_type`
Definition at line 32 of file sensor_data_type.h.
The documentation for this struct was generated from the following file:

- sensor_data_type.h

12.375 SensorIdentifier Struct Reference

Public Types

- enum **value_type** : unsigned short {
  SAMPLE_STAMP = 0 , **CHUNK_COUNT** , **IMU_VEL_X** , **IMU_VEL_Y** ,
  IMU_VEL_Z , **IMU_ACC_X** , **IMU_ACC_Y** , **IMU_ACC_Z** ,
  IMU_ANG_VEL_X , **IMU_ANG_VEL_Y** , **IMU_ANG_VEL_Z** , **IMU_ANG_ACC_X** ,
  IMU_ANG_ACC_Y , **IMU_ANG_ACC_Z** }

Static Public Member Functions

- static value_type **from_string** (const std::string &s)
- static std::string **to_string** (value_type v)
- static bool **is_valid_type** (value_type sensor_id, SensorDataType::value_type type)
- static SensorDataType::value_type **get_internal_type** (value_type sensor_id)
- static pmt::pmt_t **convert_to_internal_type** (value_type sensor_id, SensorDataType::value_type original_↩ type, const pmt::pmt_t &value)

12.375.1 Detailed Description

Definition at line 30 of file sensor_identifier.h.

12.375.2 Member Enumeration Documentation

12.375.2.1 value_type

```
enum SensorIdentifier::value_type :  unsigned short
```

Definition at line 33 of file sensor_identifier.h.
The documentation for this struct was generated from the following file:

- sensor_identifier.h

12.376 seph_t Struct Reference

Public Attributes

- int sat
- gtime_t t0
- gtime_t tof
- int sva
- int svh
- double pos [3]
- double vel [3]
- double acc [3]
- double af0
- double af1

12.376.1 Detailed Description

Definition at line 536 of file rtklib.h.

12.376.2 Member Data Documentation

12.376.2.1 acc

```
double seph_t::acc[3]
```

Definition at line 545 of file rtklib.h.

12.376.2.2 af0

```
double seph_t::af0
```

Definition at line 546 of file rtklib.h.

12.376.2.3 af1

```
double seph_t::af1
```

Definition at line 546 of file rtklib.h.

12.376.2.4 pos

```
double seph_t::pos[3]
```

Definition at line 543 of file rtklib.h.

12.376.2.5 sat

```
int seph_t::sat
```

Definition at line 538 of file rtklib.h.

12.376.2.6 sva

```
int seph_t::sva
```

Definition at line 541 of file rtklib.h.

12.376.2.7 svh

```
int seph_t::svh
```
Definition at line 542 of file rtklib.h.

12.376.2.8 t0

```
gtime_t seph_t::t0
```
Definition at line 539 of file rtklib.h.

12.376.2.9 tof

```
gtime_t seph_t::tof
```
Definition at line 540 of file rtklib.h.

12.376.2.10 vel

```
double seph_t::vel[3]
```
Definition at line 544 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.377 Serdes_Galileo_Eph Class Reference

This class implements serialization and deserialization of Galileo_Ephemeris using Protocol Buffers.
`#include <serdes_galileo_eph.h>`

Public Member Functions

- Serdes_Galileo_Eph (const Serdes_Galileo_Eph &other) noexcept
- Serdes_Galileo_Eph & operator= (const Serdes_Galileo_Eph &rhs) noexcept

  < *Copy assignment operator*
- Serdes_Galileo_Eph (Serdes_Galileo_Eph &&other) noexcept
- Serdes_Galileo_Eph & operator= (Serdes_Galileo_Eph &&other) noexcept

  < *Move assignment operator*
- std::string createProtobuffer (const std::shared_ptr< Galileo_Ephemeris > monitor)
- Galileo_Ephemeris readProtobuffer (const gnss_sdr::GalileoEphemeris &mon) const

  < *Deserialization*

12.377.1 Detailed Description

This class implements serialization and deserialization of Galileo_Ephemeris using Protocol Buffers.
Definition at line 38 of file serdes_galileo_eph.h.

12.377.2 Constructor & Destructor Documentation

12.377.2.1 Serdes_Galileo_Eph() [1/3]

```
Serdes_Galileo_Eph::Serdes_Galileo_Eph () [inline]
```
Definition at line 41 of file serdes_galileo_eph.h.

12.377.2.2 Serdes_Galileo_Eph() [2/3]

```
Serdes_Galileo_Eph::Serdes_Galileo_Eph (
            const Serdes_Galileo_Eph & other) [inline], [noexcept]
```
Definition at line 52 of file serdes_galileo_eph.h.

12.377.2.3 Serdes_Galileo_Eph() [3/3]

```
Serdes_Galileo_Eph::Serdes_Galileo_Eph (
            Serdes_Galileo_Eph && other)  [inline], [noexcept]
```

Definition at line 63 of file serdes_galileo_eph.h.

12.377.3 Member Function Documentation

12.377.3.1 createProtobuffer()

```
std::string Serdes_Galileo_Eph::createProtobuffer (
            const std::shared_ptr< Galileo_Ephemeris > monitor)  [inline]
```

Parameters

monitor	Serialization into a string

Definition at line 73 of file serdes_galileo_eph.h.

12.377.3.2 operator=() [1/2]

```
Serdes_Galileo_Eph & Serdes_Galileo_Eph::operator= (
            const Serdes_Galileo_Eph & rhs)  [inline], [noexcept]
```

< Copy assignment operator
Definition at line 56 of file serdes_galileo_eph.h.

12.377.3.3 operator=() [2/2]

```
Serdes_Galileo_Eph & Serdes_Galileo_Eph::operator= (
            Serdes_Galileo_Eph && other)  [inline], [noexcept]
```

< Move assignment operator
Definition at line 67 of file serdes_galileo_eph.h.

12.377.3.4 readProtobuffer()

```
Galileo_Ephemeris Serdes_Galileo_Eph::readProtobuffer (
            const gnss_sdr::GalileoEphemeris & mon) const  [inline]
```

< Deserialization
Definition at line 127 of file serdes_galileo_eph.h.
References Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, Galileo_Ephemeris::BGD_E1E5a, Galileo_Ephemeris::BGD_E1E5b, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::dtr, Galileo_Ephemeris::E1B_DVS, Galileo_Ephemeris::E1B_HS, Galileo_Ephemeris::E5a_DVS, Galileo_Ephemeris::E5a_HS, Galileo_Ephemeris::E5b_DVS, Galileo_Ephemeris::E5b_HS, Gnss_Ephemeris::ecc, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, Gnss_Ephemeris::M_0, Galileo_Ephemeris::nav_message_source, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdot, Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, Galileo_Ephemeris::SISA, Gnss_Ephemeris::sqrtA, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, Gnss_Ephemeris::tow, and Gnss_Ephemeris::WN.
The documentation for this class was generated from the following file:

- serdes_galileo_eph.h

12.378 Serdes_Gnss_Synchro Class Reference

This class implements serialization and deserialization of Gnss_Synchro objects using Protocol Buffers.
`#include <serdes_gnss_synchro.h>`

Public Member Functions

- Serdes_Gnss_Synchro (const Serdes_Gnss_Synchro &other) noexcept

< *Copy constructor*

- Serdes_Gnss_Synchro & operator= (const Serdes_Gnss_Synchro &rhs) noexcept

  < *Copy assignment operator*

- Serdes_Gnss_Synchro (Serdes_Gnss_Synchro &&other) noexcept

  < *Move constructor*

- Serdes_Gnss_Synchro & operator= (Serdes_Gnss_Synchro &&other) noexcept

  < *Move assignment operator*

- std::string createProtobuffer (const std::vector< Gnss_Synchro > &vgs)
- std::vector< Gnss_Synchro > readProtobuffer (const gnss_sdr::Observables &obs) const

  < *Deserialization*

12.378.1 Detailed Description

This class implements serialization and deserialization of Gnss_Synchro objects using Protocol Buffers.
Definition at line 33 of file serdes_gnss_synchro.h.

12.378.2 Constructor & Destructor Documentation

12.378.2.1 Serdes_Gnss_Synchro() [1/3]

```
Serdes_Gnss_Synchro::Serdes_Gnss_Synchro ()  [inline]
```

Definition at line 36 of file serdes_gnss_synchro.h.

12.378.2.2 Serdes_Gnss_Synchro() [2/3]

```
Serdes_Gnss_Synchro::Serdes_Gnss_Synchro (
            const Serdes_Gnss_Synchro & other)  [inline], [noexcept]
```

< Copy constructor
Definition at line 47 of file serdes_gnss_synchro.h.

12.378.2.3 Serdes_Gnss_Synchro() [3/3]

```
Serdes_Gnss_Synchro::Serdes_Gnss_Synchro (
            Serdes_Gnss_Synchro && other)  [inline], [noexcept]
```

< Move constructor
Definition at line 58 of file serdes_gnss_synchro.h.

12.378.3 Member Function Documentation

12.378.3.1 createProtobuffer()

```
std::string Serdes_Gnss_Synchro::createProtobuffer (
            const std::vector< Gnss_Synchro > & vgs)  [inline]
```

Parameters

vgs	Serialization into a string

Definition at line 72 of file serdes_gnss_synchro.h.

12.378.3.2 operator=() [1/2]

```
Serdes_Gnss_Synchro & Serdes_Gnss_Synchro::operator= (
            const Serdes_Gnss_Synchro & rhs)  [inline], [noexcept]
```

< Copy assignment operator
Definition at line 52 of file serdes_gnss_synchro.h.

12.378.3.3 operator=() [2/2]

```
Serdes_Gnss_Synchro & Serdes_Gnss_Synchro::operator= (
            Serdes_Gnss_Synchro && other)  [inline], [noexcept]
```

< Move assignment operator
Definition at line 63 of file serdes_gnss_synchro.h.

12.378.3.4 readProtobuffer()

```
std::vector< Gnss_Synchro > Serdes_Gnss_Synchro::readProtobuffer (
            const gnss_sdr::Observables & obs) const  [inline]
```

< Deserialization
Definition at line 128 of file serdes_gnss_synchro.h.
References Gnss_Synchro::Acq_delay_samples, Gnss_Synchro::Acq_doppler_hz, Gnss_Synchro::Acq_doppler_step, Gnss_Synchro::Acq_samplestamp_samples, Gnss_Synchro::Carrier_Doppler_hz, Gnss_Synchro::Carrier_phase_rads, Gnss_Synchro::Channel_ID, Gnss_Synchro::CN0_dB_hz, Gnss_Synchro::Code_phase_samples, Gnss_Synchro::correlation_length Gnss_Synchro::Flag_carrier_phase_continuous, Gnss_Synchro::Flag_cycle_slip, Gnss_Synchro::Flag_half_cycle_slip, Gnss_Synchro::Flag_PLL_180_deg_phase_locked, Gnss_Synchro::Flag_valid_acquisition, Gnss_Synchro::Flag_valid_pseudorange Gnss_Synchro::Flag_valid_symbol_output, Gnss_Synchro::Flag_valid_word, Gnss_Synchro::fs, Gnss_Synchro::Gnss_Synchro(), Gnss_Synchro::interp_TOW_ms, Gnss_Synchro::PRN, Gnss_Synchro::Prompt_I, Gnss_Synchro::Prompt_Q, Gnss_Synchro::Pseudorange_m, Gnss_Synchro::RX_time, Gnss_Synchro::Signal, Gnss_Synchro::System, Gnss_Synchro::TOW_at_current_symbol_ms, and Gnss_Synchro::Tracking_sample_counter.
The documentation for this class was generated from the following file:

- serdes_gnss_synchro.h

12.379 Serdes_Gps_Eph Class Reference

This class implements serialization and deserialization of Gps_Ephemeris objects using Protocol Buffers.
`#include <serdes_gps_eph.h>`

Public Member Functions

- Serdes_Gps_Eph (const Serdes_Gps_Eph &other) noexcept
- Serdes_Gps_Eph & operator= (const Serdes_Gps_Eph &rhs) noexcept

  < *Copy assignment operator*
- Serdes_Gps_Eph (Serdes_Gps_Eph &&other) noexcept
- Serdes_Gps_Eph & operator= (Serdes_Gps_Eph &&other) noexcept

  < *Move assignment operator*
- std::string createProtobuffer (const std::shared_ptr< Gps_Ephemeris > monitor)
- Gps_Ephemeris readProtobuffer (const gnss_sdr::GpsEphemeris &mon) const

  < *Deserialization*

12.379.1 Detailed Description

This class implements serialization and deserialization of Gps_Ephemeris objects using Protocol Buffers.
Definition at line 37 of file serdes_gps_eph.h.

12.379.2 Constructor & Destructor Documentation

12.379.2.1 Serdes_Gps_Eph() [1/3]

```
Serdes_Gps_Eph::Serdes_Gps_Eph ()  [inline]
```

Definition at line 40 of file serdes_gps_eph.h.

12.379.2.2 Serdes_Gps_Eph() [2/3]

```
Serdes_Gps_Eph::Serdes_Gps_Eph (
            const Serdes_Gps_Eph & other)  [inline], [noexcept]
```

Definition at line 51 of file serdes_gps_eph.h.

12.379.2.3 Serdes_Gps_Eph() [3/3]

```
Serdes_Gps_Eph::Serdes_Gps_Eph (
            Serdes_Gps_Eph && other)  [inline], [noexcept]
```

Definition at line 62 of file serdes_gps_eph.h.

12.379.3 Member Function Documentation

12.379.3.1 createProtobuffer()

```
std::string Serdes_Gps_Eph::createProtobuffer (
            const std::shared_ptr< Gps_Ephemeris > monitor)  [inline]
```

Parameters

monitor	Serialization into a string

Definition at line 72 of file serdes_gps_eph.h.

12.379.3.2 operator=() [1/2]

```
Serdes_Gps_Eph & Serdes_Gps_Eph::operator= (
            const Serdes_Gps_Eph & rhs)  [inline], [noexcept]
```

< Copy assignment operator
Definition at line 55 of file serdes_gps_eph.h.

12.379.3.3 operator=() [2/2]

```
Serdes_Gps_Eph & Serdes_Gps_Eph::operator= (
            Serdes_Gps_Eph && other)  [inline], [noexcept]
```

< Move assignment operator
Definition at line 66 of file serdes_gps_eph.h.

12.379.3.4 readProtobuffer()

```
Gps_Ephemeris Serdes_Gps_Eph::readProtobuffer (
            const gnss_sdr::GpsEphemeris & mon) const  [inline]
```

< Deserialization
Definition at line 127 of file serdes_gps_eph.h.
References Gnss_Ephemeris::af0, Gnss_Ephemeris::af1, Gnss_Ephemeris::af2, Gps_Ephemeris::alert_flag, Gps_Ephemeris::antispoofing_flag, Gps_Ephemeris::AODO, Gnss_Ephemeris::Cic, Gnss_Ephemeris::Cis, Gps_Ephemeris::code_on_L2, Gnss_Ephemeris::Crc, Gnss_Ephemeris::Crs, Gnss_Ephemeris::Cuc, Gnss_Ephemeris::Cus, Gnss_Ephemeris::delta_n, Gnss_Ephemeris::dtr, Gnss_Ephemeris::ecc, Gps_Ephemeris::fit_interval_flag, Gnss_Ephemeris::i_0, Gnss_Ephemeris::idot, Gps_Ephemeris::integrity_status_flag, Gps_Ephemeris::IODC, Gps_Ephemeris::IODE_SF2, Gps_Ephemeris::IODE_SF3, Gps_Ephemeris::L2_P_data_flag, Gnss_Ephemeris::M_0, Gnss_Ephemeris::omega, Gnss_Ephemeris::OMEGA_0, Gnss_Ephemeris::OMEGAdot, Gnss_Ephemeris::PRN, Gnss_Ephemeris::satClkDrift, Gnss_Ephemeris::sqrtA, Gps_Ephemeris::SV_accuracy, Gps_Ephemeris::SV_health, Gps_Ephemeris::TGD, Gnss_Ephemeris::toc, Gnss_Ephemeris::toe, Gnss_Ephemeris::tow, and Gnss_Ephemeris::WN.
The documentation for this class was generated from the following file:

- serdes_gps_eph.h

12.380 Serdes_Monitor_Pvt Class Reference

This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers.
`#include <serdes_monitor_pvt.h>`

Public Member Functions

- Serdes_Monitor_Pvt (const Serdes_Monitor_Pvt &other) noexcept
- Serdes_Monitor_Pvt & operator= (const Serdes_Monitor_Pvt &rhs) noexcept

  < *Copy assignment operator*
- Serdes_Monitor_Pvt (Serdes_Monitor_Pvt &&other) noexcept
- Serdes_Monitor_Pvt & operator= (Serdes_Monitor_Pvt &&other) noexcept

  < *Move assignment operator*
- std::string createProtobuffer (const Monitor_Pvt *const monitor)
- Monitor_Pvt readProtobuffer (const gnss_sdr::MonitorPvt &mon) const

  < *Deserialization*

12.380.1 Detailed Description

This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers.
Definition at line 38 of file serdes_monitor_pvt.h.

12.380.2 Constructor & Destructor Documentation

12.380.2.1 Serdes_Monitor_Pvt() [1/3]

```
Serdes_Monitor_Pvt::Serdes_Monitor_Pvt ()  [inline]
```
Definition at line 41 of file serdes_monitor_pvt.h.

12.380.2.2 Serdes_Monitor_Pvt() [2/3]

```
Serdes_Monitor_Pvt::Serdes_Monitor_Pvt (
            const Serdes_Monitor_Pvt & other)  [inline], [noexcept]
```
Definition at line 52 of file serdes_monitor_pvt.h.

12.380.2.3 Serdes_Monitor_Pvt() [3/3]

```
Serdes_Monitor_Pvt::Serdes_Monitor_Pvt (
            Serdes_Monitor_Pvt && other)  [inline], [noexcept]
```
Definition at line 65 of file serdes_monitor_pvt.h.

12.380.3 Member Function Documentation

12.380.3.1 createProtobuffer()

```
std::string Serdes_Monitor_Pvt::createProtobuffer (
            const Monitor_Pvt *const monitor)  [inline]
```

Parameters

monitor	Serialization into a string

Definition at line 80 of file serdes_monitor_pvt.h.

12.380.3.2 operator=() [1/2]

```
Serdes_Monitor_Pvt & Serdes_Monitor_Pvt::operator= (
            const Serdes_Monitor_Pvt & rhs)  [inline], [noexcept]
```

< Copy assignment operator
Definition at line 56 of file serdes_monitor_pvt.h.

12.380.3.3 operator=() [2/2]

```
Serdes_Monitor_Pvt & Serdes_Monitor_Pvt::operator= (
            Serdes_Monitor_Pvt && other)  [inline], [noexcept]
```

< Move assignment operator
Definition at line 71 of file serdes_monitor_pvt.h.

12.380.3.4 readProtobuffer()

```
Monitor_Pvt Serdes_Monitor_Pvt::readProtobuffer (
            const gnss_sdr::MonitorPvt & mon) const  [inline]
```

< Deserialization
Definition at line 143 of file serdes_monitor_pvt.h.
The documentation for this class was generated from the following file:

- serdes_monitor_pvt.h

12.381 Serdes_Nav_Message Class Reference

This class implements serialization and deserialization of Nav_Message_Packet objects using Protocol Buffers.
`#include <serdes_nav_message.h>`

Public Member Functions

- Serdes_Nav_Message (const Serdes_Nav_Message &other) noexcept
- Serdes_Nav_Message & operator= (const Serdes_Nav_Message &rhs) noexcept
  < *Copy assignment operator*
- Serdes_Nav_Message (Serdes_Nav_Message &&other) noexcept
- Serdes_Nav_Message & operator= (Serdes_Nav_Message &&other) noexcept
  < *Move assignment operator*
- std::string createProtobuffer (const std::shared_ptr< Nav_Message_Packet > nav_msg_packet)
- Nav_Message_Packet readProtobuffer (const gnss_sdr::navMsg &msg) const
  < *Deserialization*

12.381.1 Detailed Description

This class implements serialization and deserialization of Nav_Message_Packet objects using Protocol Buffers.
Definition at line 39 of file serdes_nav_message.h.

12.381.2 Constructor & Destructor Documentation

12.381.2.1 Serdes_Nav_Message() [1/3]

```
Serdes_Nav_Message::Serdes_Nav_Message ()  [inline]
```

Definition at line 42 of file serdes_nav_message.h.

12.381.2.2 Serdes_Nav_Message() [2/3]

```
Serdes_Nav_Message::Serdes_Nav_Message (
            const Serdes_Nav_Message & other)  [inline], [noexcept]
```

Definition at line 53 of file serdes_nav_message.h.

12.381.2.3 Serdes_Nav_Message() [3/3]

```
Serdes_Nav_Message::Serdes_Nav_Message (
            Serdes_Nav_Message && other)  [inline], [noexcept]
```

Definition at line 66 of file serdes_nav_message.h.

12.381.3 Member Function Documentation

12.381.3.1 createProtobuffer()

```
std::string Serdes_Nav_Message::createProtobuffer (
            const std::shared_ptr< Nav_Message_Packet > nav_msg_packet)  [inline]
```

Parameters

nav_msg_packet	Serialization into a string

Definition at line 82 of file serdes_nav_message.h.

12.381.3.2 operator=() [1/2]

```
Serdes_Nav_Message & Serdes_Nav_Message::operator= (
            const Serdes_Nav_Message & rhs)  [inline], [noexcept]
```

< Copy assignment operator
Definition at line 57 of file serdes_nav_message.h.

12.381.3.3 operator=() [2/2]

```
Serdes_Nav_Message & Serdes_Nav_Message::operator= (
            Serdes_Nav_Message && other)  [inline], [noexcept]
```

< Move assignment operator
Definition at line 72 of file serdes_nav_message.h.

12.381.3.4 readProtobuffer()

```
Nav_Message_Packet Serdes_Nav_Message::readProtobuffer (
            const gnss_sdr::navMsg & msg) const  [inline]
```

< Deserialization
Definition at line 101 of file serdes_nav_message.h.
References Nav_Message_Packet::nav_message, Nav_Message_Packet::prn, Nav_Message_Packet::signal, Nav_Message_Packet::system, and Nav_Message_Packet::tow_at_current_symbol_ms.
The documentation for this class was generated from the following file:

- serdes_nav_message.h

12.382 serial_t Struct Reference

Public Attributes

- dev_t dev
- int error

12.382.1 Detailed Description

Definition at line 1176 of file rtklib.h.

12.382.2 Member Data Documentation

12.382.2.1 dev

`dev_t serial_t::dev`
Definition at line 1178 of file rtklib.h.

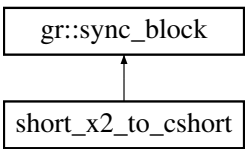
12.382.2.2 error

`int serial_t::error`
Definition at line 1179 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.383 short_x2_to_cshort Class Reference

This class adapts two short streams into a std::complex<short> stream.
`#include <short_x2_to_cshort.h>`
Inheritance diagram for short_x2_to_cshort:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- short_x2_to_cshort_sptr **make_short_x2_to_cshort** ()

12.383.1 Detailed Description

This class adapts two short streams into a std::complex<short> stream.
Definition at line 40 of file short_x2_to_cshort.h.
The documentation for this class was generated from the following file:

- short_x2_to_cshort.h

12.384 Signal_Enabled_Flags Class Reference

Public Member Functions

- **Signal_Enabled_Flags** (const ConfigurationInterface *configuration)
- **Signal_Enabled_Flags** (uint32_t flags_)
- template<typename T>
  uint32_t or_all (const T &value) const
- template<typename T, typename... Args>
  uint32_t or_all (const T &first, const Args &... rest) const
- template<typename... Args>
  bool check_only_enabled (const Args &... args) const
- template<typename... Args>
  bool check_any_enabled (const Args &... args) const

Public Attributes

- const uint32_t flags
- const bool has_gps
- const bool has_galileo
- const bool has_glonass
- const bool has_beidou
- const bool has_qzss
- const bool only_gps
- const bool only_galileo
- const bool only_glonass
- const bool only_beidou
- const bool only_qzss

12.384.1 Detailed Description

Definition at line 25 of file signal_enabled_flags.h.

12.384.2 Member Function Documentation

12.384.2.1 check_any_enabled()

```
template<typename... Args>
bool Signal_Enabled_Flags::check_any_enabled (
            const Args &... args) const [inline]
```

Definition at line 50 of file signal_enabled_flags.h.

12.384.2.2 check_only_enabled()

```
template<typename... Args>
bool Signal_Enabled_Flags::check_only_enabled (
            const Args &... args) const [inline]
```

Definition at line 44 of file signal_enabled_flags.h.

12.384.2.3 or_all() [1/2]

```
template<typename T, typename... Args>
uint32_t Signal_Enabled_Flags::or_all (
            const T & first,
            const Args &... rest) const [inline]
```

Definition at line 38 of file signal_enabled_flags.h.

12.384.2.4 or_all() [2/2]

```
template<typename T>
uint32_t Signal_Enabled_Flags::or_all (
            const T & value) const [inline]
```

Definition at line 32 of file signal_enabled_flags.h.

12.384.3 Member Data Documentation

12.384.3.1 flags

```
const uint32_t Signal_Enabled_Flags::flags
```

Definition at line 55 of file signal_enabled_flags.h.

12.384.3.2 has_beidou

```
const bool Signal_Enabled_Flags::has_beidou
```

Definition at line 60 of file signal_enabled_flags.h.

12.384.3.3 has_galileo

`const bool Signal_Enabled_Flags::has_galileo`
Definition at line 58 of file signal_enabled_flags.h.

12.384.3.4 has_glonass

`const bool Signal_Enabled_Flags::has_glonass`
Definition at line 59 of file signal_enabled_flags.h.

12.384.3.5 has_gps

`const bool Signal_Enabled_Flags::has_gps`
Definition at line 57 of file signal_enabled_flags.h.

12.384.3.6 has_qzss

`const bool Signal_Enabled_Flags::has_qzss`
Definition at line 61 of file signal_enabled_flags.h.

12.384.3.7 only_beidou

`const bool Signal_Enabled_Flags::only_beidou`
Definition at line 66 of file signal_enabled_flags.h.

12.384.3.8 only_galileo

`const bool Signal_Enabled_Flags::only_galileo`
Definition at line 64 of file signal_enabled_flags.h.

12.384.3.9 only_glonass

`const bool Signal_Enabled_Flags::only_glonass`
Definition at line 65 of file signal_enabled_flags.h.

12.384.3.10 only_gps

`const bool Signal_Enabled_Flags::only_gps`
Definition at line 63 of file signal_enabled_flags.h.

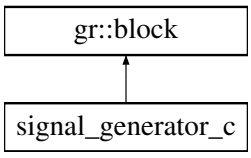
12.384.3.11 only_qzss

`const bool Signal_Enabled_Flags::only_qzss`
Definition at line 67 of file signal_enabled_flags.h.
The documentation for this class was generated from the following file:

- signal_enabled_flags.h

12.385 signal_generator_c Class Reference

This class generates synthesized GNSS signal.
`#include <signal_generator_c.h>`
Inheritance diagram for signal_generator_c:



Public Member Functions

- int **general_work** (int noutput_items, gr_vector_int &ninput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- signal_generator_c_sptr signal_make_generator_c (const std::vector< std::string > &signal1, const std::vector< std::string > &system, const std::vector< unsigned int > &PRN, const std::vector< float > &CN0_dB, const std::vector< float > &doppler_Hz, const std::vector< unsigned int > &delay_chips, const std::vector< unsigned int > &delay_sec, bool data_flag, bool noise_flag, unsigned int fs_in, unsigned int vector_length, float BW_BB)

  Return a shared_ptr to a new instance of gen_source.

12.385.1 Detailed Description

This class generates synthesized GNSS signal.

See also

gen_source for a version that subclasses gr_block.

Definition at line 58 of file signal_generator_c.h.

12.385.2 Friends And Related Symbol Documentation

12.385.2.1 signal_make_generator_c

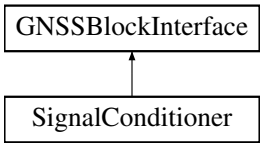
```
signal_generator_c_sptr signal_make_generator_c (
            const std::vector< std::string > & signal1,
            const std::vector< std::string > & system,
            const std::vector< unsigned int > & PRN,
            const std::vector< float > & CN0_dB,
            const std::vector< float > & doppler_Hz,
            const std::vector< unsigned int > & delay_chips,
            const std::vector< unsigned int > & delay_sec,
            bool data_flag,
            bool noise_flag,
            unsigned int fs_in,
            unsigned int vector_length,
            float BW_BB)  [friend]
```

Return a shared_ptr to a new instance of gen_source.
To avoid accidental use of raw pointers, gen_source's constructor is private. signal_make_generator_c is the public interface for creating new instances.
The documentation for this class was generated from the following file:

- signal_generator_c.h

12.386 SignalConditioner Class Reference

This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.
`#include <signal_conditioner.h>`
Inheritance diagram for SignalConditioner:



Public Member Functions

- SignalConditioner (std::shared_ptr< GNSSBlockInterface > data_type_adapt, std::shared_ptr< GNSSBlockInterface > in_filt, std::shared_ptr< GNSSBlockInterface > res, std::string role)

  Constructor.
- ~SignalConditioner ()=default

  Destructor.
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string role () override
- std::string implementation () override

  Returns "Signal_Conditioner".
- size_t item_size () override
- std::shared_ptr< GNSSBlockInterface > data_type_adapter ()
- std::shared_ptr< GNSSBlockInterface > input_filter ()
- std::shared_ptr< GNSSBlockInterface > resampler ()

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.386.1 Detailed Description

This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.
Definition at line 39 of file signal_conditioner.h.

12.386.2 Constructor & Destructor Documentation

12.386.2.1 SignalConditioner()

```
SignalConditioner::SignalConditioner (
            std::shared_ptr< GNSSBlockInterface > data_type_adapt,
            std::shared_ptr< GNSSBlockInterface > in_filt,
            std::shared_ptr< GNSSBlockInterface > res,
            std::string role)
```

Constructor.

12.386.2.2 ~SignalConditioner()

```
SignalConditioner::~SignalConditioner ()  [default]
```

Destructor.

12.386.3 Member Function Documentation

12.386.3.1 connect()

```
void SignalConditioner::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.386.3.2 data_type_adapter()

```
std::shared_ptr< GNSSBlockInterface > SignalConditioner::data_type_adapter ()  [inline]
```
Definition at line 62 of file signal_conditioner.h.

12.386.3.3 disconnect()

```
void SignalConditioner::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```
Implements GNSSBlockInterface.

12.386.3.4 get_left_block()

```
gr::basic_block_sptr SignalConditioner::get_left_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.386.3.5 get_right_block()

```
gr::basic_block_sptr SignalConditioner::get_right_block ()  [override], [virtual]
```
Implements GNSSBlockInterface.

12.386.3.6 implementation()

```
std::string SignalConditioner::implementation ()  [inline], [override], [virtual]
```
Returns "Signal_Conditioner".
Implements GNSSBlockInterface.
Definition at line 58 of file signal_conditioner.h.

12.386.3.7 input_filter()

```
std::shared_ptr< GNSSBlockInterface > SignalConditioner::input_filter ()  [inline]
```
Definition at line 63 of file signal_conditioner.h.

12.386.3.8 item_size()

```
size_t SignalConditioner::item_size ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 60 of file signal_conditioner.h.

12.386.3.9 resampler()

```
std::shared_ptr< GNSSBlockInterface > SignalConditioner::resampler ()  [inline]
```
Definition at line 64 of file signal_conditioner.h.

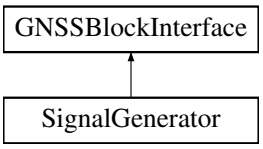
12.386.3.10 role()

```
std::string SignalConditioner::role ()  [inline], [override], [virtual]
```
Implements GNSSBlockInterface.
Definition at line 56 of file signal_conditioner.h.
The documentation for this class was generated from the following file:

- signal_conditioner.h

12.387 SignalGenerator Class Reference

This class generates synthesized GNSS signal.
`#include <signal_generator.h>`
Inheritance diagram for SignalGenerator:



Public Member Functions

- **SignalGenerator** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- std::string role () override
- std::string implementation () override

  Returns "GNSSSignalGenerator".
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.387.1 Detailed Description

This class generates synthesized GNSS signal.
Definition at line 39 of file signal_generator.h.

12.387.2 Member Function Documentation

12.387.2.1 connect()

```
void SignalGenerator::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.387.2.2 disconnect()

```
void SignalGenerator::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.387.2.3 get_left_block()

```
gr::basic_block_sptr SignalGenerator::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.387.2.4 get_right_block()

```
gr::basic_block_sptr SignalGenerator::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.387.2.5 implementation()

`std::string SignalGenerator::implementation () [inline], [override], [virtual]`
Returns "GNSSSignalGenerator".
Implements GNSSBlockInterface.
Definition at line 56 of file signal_generator.h.

12.387.2.6 item_size()

`size_t SignalGenerator::item_size () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 61 of file signal_generator.h.

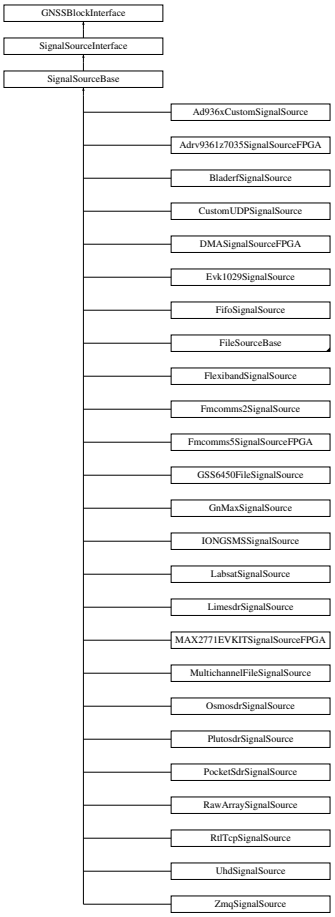
12.387.2.7 role()

`std::string SignalGenerator::role () [inline], [override], [virtual]`
Implements GNSSBlockInterface.
Definition at line 48 of file signal_generator.h.
The documentation for this class was generated from the following file:

- signal_generator.h

12.388 SignalSourceBase Class Reference

Inheritance diagram for SignalSourceBase:



Public Member Functions

- std::string role () final

- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.388.1 Detailed Description

Definition at line 28 of file signal_source_base.h.

12.388.2 Constructor & Destructor Documentation

12.388.2.1 SignalSourceBase()

```
SignalSourceBase::SignalSourceBase (
            ConfigurationInterface const * configuration,
            std::string role,
            std::string impl)  [protected]
```

Constructor.

12.388.3 Member Function Documentation

12.388.3.1 decode_item_type()

```
size_t SignalSourceBase::decode_item_type (
            std::string const & item_type,
            bool * is_interleaved = nullptr,
            bool throw_on_error = false)  [protected]
```

utility for decoding passed ".item_type" values

Parameters

in	*item_type*	- user provided string, should be one of the known types
out	*is_interleaved*	- if non-null, the pointed to memory is updated with whether the data is interleaved I/Q (e.g., ishort)
in	*throw_on_error*	- if true, throw an exception if the string does not represent a known type

Returns

the size in bytes of the passed type

12.388.3.2 get_left_block()

`gr::basic_block_sptr SignalSourceBase::get_left_block () [override], [virtual]`
Implements GNSSBlockInterface.

12.388.3.3 getRfChannels()

`size_t SignalSourceBase::getRfChannels () const [override], [virtual]`
Implements SignalSourceInterface.

12.388.3.4 implementation()

`std::string SignalSourceBase::implementation () [final], [virtual]`
Implements GNSSBlockInterface.

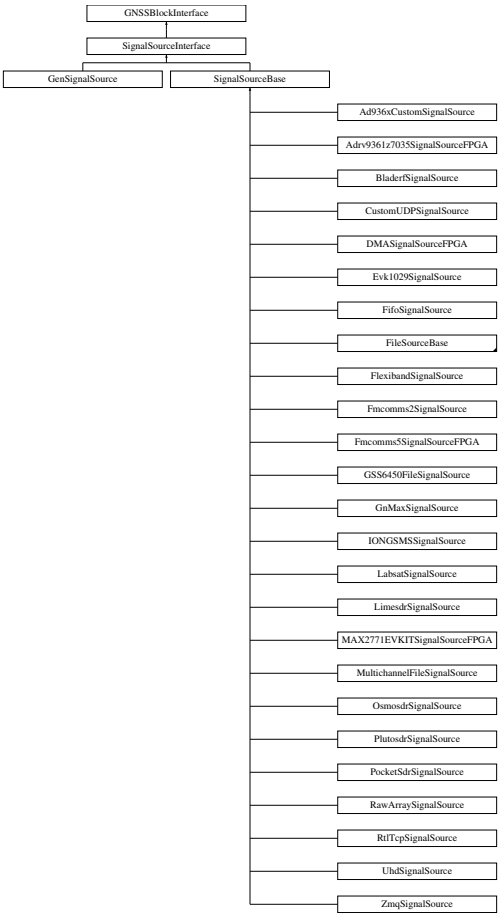
12.388.3.5 role()

`std::string SignalSourceBase::role () [final], [virtual]`
Implements GNSSBlockInterface.
The documentation for this class was generated from the following file:

- signal_source_base.h

12.389 SignalSourceInterface Class Reference

This abstract class represents an interface to signal_source GNSS block.
`#include <signal_source_interface.h>`
Inheritance diagram for SignalSourceInterface:



Public Member Functions

- virtual size_t **getRfChannels** () const =0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.389.1 Detailed Description

This abstract class represents an interface to signal_source GNSS block.
Abstract class for signal sources. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 51 of file signal_source_interface.h.
The documentation for this class was generated from the following file:

- signal_source_interface.h

12.390 snrmask_t Struct Reference

Public Attributes

- int ena [2]
- double mask [NFREQ][9]

12.390.1 Detailed Description

Definition at line 975 of file rtklib.h.

12.390.2 Member Data Documentation

12.390.2.1 ena

```
int snrmask_t::ena[2]
```

Definition at line 977 of file rtklib.h.

12.390.2.2 mask

```
double snrmask_t::mask[NFREQ][9]
```

Definition at line 978 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.391 sol_t Struct Reference

Public Attributes

- gtime_t time
- double rr [6]
- float qr [6]
- double dtr [6]
- unsigned char type
- unsigned char stat
- unsigned char ns
- float age
- float ratio
- float thres
- float prev_ratio1
- float prev_ratio2

12.391.1 Detailed Description

Definition at line 857 of file rtklib.h.

12.391.2 Member Data Documentation

12.391.2.1 age

```
float sol_t::age
```

Definition at line 869 of file rtklib.h.

12.391.2.2 dtr

```
double sol_t::dtr[6]
```

Definition at line 865 of file rtklib.h.

12.391.2.3 ns

`unsigned char sol_t::ns`
Definition at line 868 of file rtklib.h.

12.391.2.4 prev_ratio1

`float sol_t::prev_ratio1`
Definition at line 872 of file rtklib.h.

12.391.2.5 prev_ratio2

`float sol_t::prev_ratio2`
Definition at line 873 of file rtklib.h.

12.391.2.6 qr

`float sol_t::qr[6]`
Definition at line 862 of file rtklib.h.

12.391.2.7 ratio

`float sol_t::ratio`
Definition at line 870 of file rtklib.h.

12.391.2.8 rr

`double sol_t::rr[6]`
Definition at line 860 of file rtklib.h.

12.391.2.9 stat

`unsigned char sol_t::stat`
Definition at line 867 of file rtklib.h.

12.391.2.10 thres

`float sol_t::thres`
Definition at line 871 of file rtklib.h.

12.391.2.11 time

`gtime_t sol_t::time`
Definition at line 859 of file rtklib.h.

12.391.2.12 type

`unsigned char sol_t::type`
Definition at line 866 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.392 solbuf_t Struct Reference

Public Attributes

- int n
- int nmax
- int cyclic

- int start
- int end
- gtime_t time
- sol_t * data
- double rb [3]
- unsigned char buff [MAXSOLMSG+1]
- int nb

12.392.1 Detailed Description

Definition at line 877 of file rtklib.h.

12.392.2 Member Data Documentation

12.392.2.1 buff

```
unsigned char solbuf_t::buff[MAXSOLMSG+1]
```

Definition at line 885 of file rtklib.h.

12.392.2.2 cyclic

```
int solbuf_t::cyclic
```

Definition at line 880 of file rtklib.h.

12.392.2.3 data

```
sol_t* solbuf_t::data
```

Definition at line 883 of file rtklib.h.

12.392.2.4 end

```
int solbuf_t::end
```

Definition at line 881 of file rtklib.h.

12.392.2.5 n

```
int solbuf_t::n
```

Definition at line 879 of file rtklib.h.

12.392.2.6 nb

```
int solbuf_t::nb
```

Definition at line 886 of file rtklib.h.

12.392.2.7 nmax

```
int solbuf_t::nmax
```

Definition at line 879 of file rtklib.h.

12.392.2.8 rb

```
double solbuf_t::rb[3]
```

Definition at line 884 of file rtklib.h.

12.392.2.9 start

```
int solbuf_t::start
```

Definition at line 881 of file rtklib.h.

12.392.2.10 time

`gtime_t solbuf_t::time`
Definition at line 882 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.393 solopt_t Struct Reference

Public Attributes

- int posf
- int times
- int timef
- int timeu
- int degf
- int outhead
- int outopt
- int datum
- int height
- int geoid
- int solstatic
- int sstat
- int trace
- double nmeaintv [2]
- char sep [64]
- char prog [64]
- double maxsolstd

12.393.1 Detailed Description

Definition at line 1067 of file rtklib.h.

12.393.2 Member Data Documentation

12.393.2.1 datum

`int solopt_t::datum`
Definition at line 1076 of file rtklib.h.

12.393.2.2 degf

`int solopt_t::degf`
Definition at line 1073 of file rtklib.h.

12.393.2.3 geoid

`int solopt_t::geoid`
Definition at line 1078 of file rtklib.h.

12.393.2.4 height

`int solopt_t::height`
Definition at line 1077 of file rtklib.h.

12.393.2.5 maxsolstd

`double solopt_t::maxsolstd`
Definition at line 1086 of file rtklib.h.

12.393.2.6 nmeaintv

```
double solopt_t::nmeaintv[2]
```
Definition at line 1082 of file rtklib.h.

12.393.2.7 outhead

```
int solopt_t::outhead
```
Definition at line 1074 of file rtklib.h.

12.393.2.8 outopt

```
int solopt_t::outopt
```
Definition at line 1075 of file rtklib.h.

12.393.2.9 posf

```
int solopt_t::posf
```
Definition at line 1069 of file rtklib.h.

12.393.2.10 prog

```
char solopt_t::prog[64]
```
Definition at line 1085 of file rtklib.h.

12.393.2.11 sep

```
char solopt_t::sep[64]
```
Definition at line 1084 of file rtklib.h.

12.393.2.12 solstatic

```
int solopt_t::solstatic
```
Definition at line 1079 of file rtklib.h.

12.393.2.13 sstat

```
int solopt_t::sstat
```
Definition at line 1080 of file rtklib.h.

12.393.2.14 timef

```
int solopt_t::timef
```
Definition at line 1071 of file rtklib.h.

12.393.2.15 times

```
int solopt_t::times
```
Definition at line 1070 of file rtklib.h.

12.393.2.16 timeu

```
int solopt_t::timeu
```
Definition at line 1072 of file rtklib.h.

12.393.2.17 trace

```
int solopt_t::trace
```
Definition at line 1081 of file rtklib.h.

The documentation for this struct was generated from the following file:

- rtklib.h

12.394 solstat_t Struct Reference

Public Attributes

- gtime_t time
- unsigned char sat
- unsigned char frq
- float az
- float el
- float resp
- float resc
- unsigned char flag
- unsigned char snr
- unsigned short lock
- unsigned short outc
- unsigned short slipc
- unsigned short rejc

12.394.1 Detailed Description

Definition at line 890 of file rtklib.h.

12.394.2 Member Data Documentation

12.394.2.1 az

```
float solstat_t::az
```

Definition at line 895 of file rtklib.h.

12.394.2.2 el

```
float solstat_t::el
```

Definition at line 895 of file rtklib.h.

12.394.2.3 flag

```
unsigned char solstat_t::flag
```

Definition at line 898 of file rtklib.h.

12.394.2.4 frq

```
unsigned char solstat_t::frq
```

Definition at line 894 of file rtklib.h.

12.394.2.5 lock

```
unsigned short solstat_t::lock
```

Definition at line 900 of file rtklib.h.

12.394.2.6 outc

```
unsigned short solstat_t::outc
```

Definition at line 901 of file rtklib.h.

12.394.2.7 rejc

```
unsigned short solstat_t::rejc
```

Definition at line 903 of file rtklib.h.

12.394.2.8 resc

```
float solstat_t::resc
```
Definition at line 897 of file rtklib.h.

12.394.2.9 resp

```
float solstat_t::resp
```
Definition at line 896 of file rtklib.h.

12.394.2.10 sat

```
unsigned char solstat_t::sat
```
Definition at line 893 of file rtklib.h.

12.394.2.11 slipc

```
unsigned short solstat_t::slipc
```
Definition at line 902 of file rtklib.h.

12.394.2.12 snr

```
unsigned char solstat_t::snr
```
Definition at line 899 of file rtklib.h.

12.394.2.13 time

```
gtime_t solstat_t::time
```
Definition at line 892 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.395 solstatbuf_t Struct Reference

Public Attributes

- int n
- int nmax
- solstat_t * data

12.395.1 Detailed Description

Definition at line 907 of file rtklib.h.

12.395.2 Member Data Documentation

12.395.2.1 data

```
solstat_t* solstatbuf_t::data
```
Definition at line 910 of file rtklib.h.

12.395.2.2 n

```
int solstatbuf_t::n
```
Definition at line 909 of file rtklib.h.

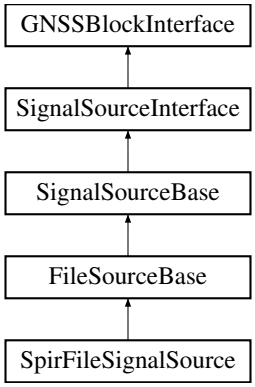
12.395.2.3 nmax

`int solstatbuf_t::nmax`
Definition at line 909 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.396 SpirFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <spir_file_signal_source.h>`
Inheritance diagram for SpirFileSignalSource:



Public Member Functions

- **SpirFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in↩ _streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void post_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual double packetsPerSample () const

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **pre_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.396.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 40 of file spir_file_signal_source.h.

12.396.2 Member Function Documentation

12.396.2.1 create_file_source_hook()

```
void SpirFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.396.2.2 itemTypeToSize()

```
std::tuple< size_t, bool > SpirFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```

Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.396.2.3 post_disconnect_hook()

```
void SpirFileSignalSource::post_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.396.2.4 pre_connect_hook()

```
void SpirFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.396.2.5 source()

```
gnss_shared_ptr< gr::block > SpirFileSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- spir_file_signal_source.h

12.397 ssat_t Struct Reference

Public Attributes

- unsigned char sys
- unsigned char vs
- double azel [2]
- double resp [NFREQ]
- double resc [NFREQ]

- unsigned char vsat [NFREQ]
- unsigned char snr [NFREQ]
- unsigned char code [NFREQ]
- unsigned char fix [NFREQ]
- unsigned char slip [NFREQ]
- unsigned char half [NFREQ]
- int lock [NFREQ]
- unsigned int outc [NFREQ]
- unsigned int slipc [NFREQ]
- unsigned int rejc [NFREQ]
- double gf
- double gf2
- double mw
- double phw
- gtime_t pt [2][NFREQ]
- double ph [2][NFREQ]
- float snr_rover [NFREQ]
- float snr_base [NFREQ]

12.397.1 Detailed Description

Definition at line 1090 of file rtklib.h.

12.397.2 Member Data Documentation

12.397.2.1 azel

```
double ssat_t::azel[2]
```

Definition at line 1094 of file rtklib.h.

12.397.2.2 code

```
unsigned char ssat_t::code[NFREQ]
```

Definition at line 1099 of file rtklib.h.

12.397.2.3 fix

```
unsigned char ssat_t::fix[NFREQ]
```

Definition at line 1100 of file rtklib.h.

12.397.2.4 gf

```
double ssat_t::gf
```

Definition at line 1107 of file rtklib.h.

12.397.2.5 gf2

```
double ssat_t::gf2
```

Definition at line 1108 of file rtklib.h.

12.397.2.6 half

```
unsigned char ssat_t::half[NFREQ]
```

Definition at line 1102 of file rtklib.h.

12.397.2.7 lock

```
int ssat_t::lock[NFREQ]
```

Definition at line 1103 of file rtklib.h.

12.397.2.8 mw

```
double ssat_t::mw
```
Definition at line 1109 of file rtklib.h.

12.397.2.9 outc

```
unsigned int ssat_t::outc[NFREQ]
```
Definition at line 1104 of file rtklib.h.

12.397.2.10 ph

```
double ssat_t::ph[2][NFREQ]
```
Definition at line 1112 of file rtklib.h.

12.397.2.11 phw

```
double ssat_t::phw
```
Definition at line 1110 of file rtklib.h.

12.397.2.12 pt

```
gtime_t ssat_t::pt[2][NFREQ]
```
Definition at line 1111 of file rtklib.h.

12.397.2.13 rejc

```
unsigned int ssat_t::rejc[NFREQ]
```
Definition at line 1106 of file rtklib.h.

12.397.2.14 resc

```
double ssat_t::resc[NFREQ]
```
Definition at line 1096 of file rtklib.h.

12.397.2.15 resp

```
double ssat_t::resp[NFREQ]
```
Definition at line 1095 of file rtklib.h.

12.397.2.16 slip

```
unsigned char ssat_t::slip[NFREQ]
```
Definition at line 1101 of file rtklib.h.

12.397.2.17 slipc

```
unsigned int ssat_t::slipc[NFREQ]
```
Definition at line 1105 of file rtklib.h.

12.397.2.18 snr

```
unsigned char ssat_t::snr[NFREQ]
```
Definition at line 1098 of file rtklib.h.

12.397.2.19 snr_base

```
float ssat_t::snr_base[NFREQ]
```
Definition at line 1114 of file rtklib.h.

12.397.2.20 snr_rover

```
float ssat_t::snr_rover[NFREQ]
```
Definition at line 1113 of file rtklib.h.

12.397.2.21 sys

```
unsigned char ssat_t::sys
```
Definition at line 1092 of file rtklib.h.

12.397.2.22 vs

```
unsigned char ssat_t::vs
```
Definition at line 1093 of file rtklib.h.

12.397.2.23 vsat

```
unsigned char ssat_t::vsat[NFREQ]
```
Definition at line 1097 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.398 ssr_t Struct Reference

Public Attributes

- gtime_t t0 [6]
- double udi [6]
- int iod [6]
- int iode
- int iodcrc
- int ura
- int refd
- double deph [3]
- double ddeph [3]
- double dclk [3]
- double hrclk
- float cbias [MAXCODE]
- double pbias [MAXCODE]
- float stdpb [MAXCODE]
- double yaw_ang
- double yaw_rate
- unsigned char update

12.398.1 Detailed Description

Definition at line 691 of file rtklib.h.

12.398.2 Member Data Documentation

12.398.2.1 cbias

```
float ssr_t::cbias[MAXCODE]
```
Definition at line 704 of file rtklib.h.

12.398.2.2 dclk

```
double ssr_t::dclk[3]
```
Definition at line 702 of file rtklib.h.

12.398.2.3 ddeph

`double ssr_t::ddeph[3]`
Definition at line 701 of file rtklib.h.

12.398.2.4 deph

`double ssr_t::deph[3]`
Definition at line 700 of file rtklib.h.

12.398.2.5 hrclk

`double ssr_t::hrclk`
Definition at line 703 of file rtklib.h.

12.398.2.6 iod

`int ssr_t::iod[6]`
Definition at line 695 of file rtklib.h.

12.398.2.7 iodcrc

`int ssr_t::iodcrc`
Definition at line 697 of file rtklib.h.

12.398.2.8 iode

`int ssr_t::iode`
Definition at line 696 of file rtklib.h.

12.398.2.9 pbias

`double ssr_t::pbias[MAXCODE]`
Definition at line 705 of file rtklib.h.

12.398.2.10 refd

`int ssr_t::refd`
Definition at line 699 of file rtklib.h.

12.398.2.11 stdpb

`float ssr_t::stdpb[MAXCODE]`
Definition at line 706 of file rtklib.h.

12.398.2.12 t0

`gtime_t ssr_t::t0[6]`
Definition at line 693 of file rtklib.h.

12.398.2.13 udi

`double ssr_t::udi[6]`
Definition at line 694 of file rtklib.h.

12.398.2.14 update

`unsigned char ssr_t::update`
Definition at line 708 of file rtklib.h.

12.398.2.15 ura

```
int ssr_t::ura
```
Definition at line 698 of file rtklib.h.

12.398.2.16 yaw_ang

```
double ssr_t::yaw_ang
```
Definition at line 707 of file rtklib.h.

12.398.2.17 yaw_rate

```
double ssr_t::yaw_rate
```
Definition at line 707 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.399 sta_t Struct Reference

Public Attributes

- char name [MAXANT]
- char marker [MAXANT]
- char antdes [MAXANT]
- char antsno [MAXANT]
- char rectype [MAXANT]
- char recver [MAXANT]
- char recsno [MAXANT]
- int antsetup
- int itrf
- int deltype
- double pos [3]
- double del [3]
- double hgt

12.399.1 Detailed Description

Definition at line 839 of file rtklib.h.

12.399.2 Member Data Documentation

12.399.2.1 antdes

```
char sta_t::antdes[MAXANT]
```
Definition at line 843 of file rtklib.h.

12.399.2.2 antsetup

```
int sta_t::antsetup
```
Definition at line 848 of file rtklib.h.

12.399.2.3 antsno

```
char sta_t::antsno[MAXANT]
```
Definition at line 844 of file rtklib.h.

12.399.2.4 del

`double sta_t::del[3]`
Definition at line 852 of file rtklib.h.

12.399.2.5 deltype

`int sta_t::deltype`
Definition at line 850 of file rtklib.h.

12.399.2.6 hgt

`double sta_t::hgt`
Definition at line 853 of file rtklib.h.

12.399.2.7 itrf

`int sta_t::itrf`
Definition at line 849 of file rtklib.h.

12.399.2.8 marker

`char sta_t::marker[MAXANT]`
Definition at line 842 of file rtklib.h.

12.399.2.9 name

`char sta_t::name[MAXANT]`
Definition at line 841 of file rtklib.h.

12.399.2.10 pos

`double sta_t::pos[3]`
Definition at line 851 of file rtklib.h.

12.399.2.11 recsno

`char sta_t::recsno[MAXANT]`
Definition at line 847 of file rtklib.h.

12.399.2.12 rectype

`char sta_t::rectype[MAXANT]`
Definition at line 845 of file rtklib.h.

12.399.2.13 recver

`char sta_t::recver[MAXANT]`
Definition at line 846 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.400 stec_t Struct Reference

Public Attributes

- gtime_t time
- unsigned char sat
- double ion

- float std
- float azel [2]
- unsigned char flag

12.400.1 Detailed Description

Definition at line 757 of file rtklib.h.

12.400.2 Member Data Documentation

12.400.2.1 azel

```
float stec_t::azel[2]
```
Definition at line 763 of file rtklib.h.

12.400.2.2 flag

```
unsigned char stec_t::flag
```
Definition at line 764 of file rtklib.h.

12.400.2.3 ion

```
double stec_t::ion
```
Definition at line 761 of file rtklib.h.

12.400.2.4 sat

```
unsigned char stec_t::sat
```
Definition at line 760 of file rtklib.h.

12.400.2.5 std

```
float stec_t::std
```
Definition at line 762 of file rtklib.h.

12.400.2.6 time

```
gtime_t stec_t::time
```
Definition at line 759 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.401 stream_cfg Struct Reference

Public Attributes

- int64_t bw_hz
- int64_t fs_hz
- int64_t lo_hz
- const char * rfport

12.401.1 Detailed Description

Definition at line 50 of file ad9361_manager.h.

12.401.2 Member Data Documentation

12.401.2.1 bw_hz

```
int64_t stream_cfg::bw_hz
```
Definition at line 52 of file ad9361_manager.h.

12.401.2.2 fs_hz

```
int64_t stream_cfg::fs_hz
```
Definition at line 53 of file ad9361_manager.h.

12.401.2.3 lo_hz

```
int64_t stream_cfg::lo_hz
```
Definition at line 54 of file ad9361_manager.h.

12.401.2.4 rfport

```
const char* stream_cfg::rfport
```
Definition at line 55 of file ad9361_manager.h.
The documentation for this struct was generated from the following file:

- ad9361_manager.h

12.402 stream_t Struct Reference

Public Attributes

- int type
- int mode
- int state
- unsigned int inb
- unsigned int inr
- unsigned int outb
- unsigned int outr
- unsigned int tick
- unsigned int tact
- unsigned int inbt
- unsigned int outbt
- lock_t lock
- void * port
- char path [MAXSTRPATH]
- char msg [MAXSTRMSG]

12.402.1 Detailed Description

Definition at line 1160 of file rtklib.h.

12.402.2 Member Data Documentation

12.402.2.1 inb

```
unsigned int stream_t::inb
```
Definition at line 1165 of file rtklib.h.

12.402.2.2 inbt

```
unsigned int stream_t::inbt
```
Definition at line 1168 of file rtklib.h.

12.402.2.3 inr

```
unsigned int stream_t::inr
```
Definition at line 1165 of file rtklib.h.

12.402.2.4 lock

```
lock_t stream_t::lock
```
Definition at line 1169 of file rtklib.h.

12.402.2.5 mode

```
int stream_t::mode
```
Definition at line 1163 of file rtklib.h.

12.402.2.6 msg

```
char stream_t::msg[MAXSTRMSG]
```
Definition at line 1172 of file rtklib.h.

12.402.2.7 outb

```
unsigned int stream_t::outb
```
Definition at line 1166 of file rtklib.h.

12.402.2.8 outbt

```
unsigned int stream_t::outbt
```
Definition at line 1168 of file rtklib.h.

12.402.2.9 outr

```
unsigned int stream_t::outr
```
Definition at line 1166 of file rtklib.h.

12.402.2.10 path

```
char stream_t::path[MAXSTRPATH]
```
Definition at line 1171 of file rtklib.h.

12.402.2.11 port

```
void* stream_t::port
```
Definition at line 1170 of file rtklib.h.

12.402.2.12 state

```
int stream_t::state
```
Definition at line 1164 of file rtklib.h.

12.402.2.13 tact

```
unsigned int stream_t::tact
```
Definition at line 1167 of file rtklib.h.

12.402.2.14 tick

```
unsigned int stream_t::tick
```
Definition at line 1167 of file rtklib.h.

12.402.2.15 type

```
int stream_t::type
```

Definition at line 1162 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.403 StringConverter Class Reference

Class that interprets the contents of a string and converts it into different types.
`#include <string_converter.h>`

Public Member Functions

- bool **convert** (const std::string &value, bool default_value)
- int64_t **convert** (const std::string &value, int64_t default_value)
- uint64_t **convert** (const std::string &value, uint64_t default_value)
- int32_t **convert** (const std::string &value, int32_t default_value)
- uint32_t **convert** (const std::string &value, uint32_t default_value)
- int16_t **convert** (const std::string &value, int16_t default_value)
- uint16_t **convert** (const std::string &value, uint16_t default_value)
- float **convert** (const std::string &value, float default_value)
- double **convert** (const std::string &value, double default_value)

12.403.1 Detailed Description

Class that interprets the contents of a string and converts it into different types.
Definition at line 35 of file string_converter.h.
The documentation for this class was generated from the following file:

- string_converter.h

12.404 Tag Class Reference

Public Types

- enum **e_verification_status** { **SUCCESS** , **FAIL** , **UNVERIFIED** , **AUTHENTICATED_DONT_USE** }

Public Member Functions

- Tag (const MACK_tag_and_info &MTI, uint32_t TOW, uint32_t WN, uint32_t PRNa, uint8_t CTR, uint8_t nmas, std::vector< uint8_t > allowed_adkds={0, 4, 12})
- Tag (const MACK_message &mack)

Public Attributes

- const uint32_t tag_id
- uint32_t TOW
- uint32_t WN
- uint32_t PRNa
- uint8_t CTR
- uint8_t nmas
- e_verification_status status
- uint64_t received_tag
- uint64_t computed_tag
- uint8_t PRN_d

- uint8_t ADKD
- uint8_t cop
- uint32_t skipped
- std::vector< uint8_t > allowed_adkds {0, 4, 12}
- std::string nav_data

Static Public Attributes

- static uint32_t id_counter

12.404.1 Detailed Description

Definition at line 251 of file osnma_data.h.

12.404.2 Member Enumeration Documentation

12.404.2.1 e_verification_status

```
enum Tag::e_verification_status
```

Definition at line 254 of file osnma_data.h.

12.404.3 Constructor & Destructor Documentation

12.404.3.1 Tag() [1/2]

```
Tag::Tag (
            const MACK_tag_and_info & MTI,
            uint32_t TOW,
            uint32_t WN,
            uint32_t PRNa,
            uint8_t CTR,
            uint8_t nmas,
            std::vector< uint8_t > allowed_adkds = {0, 4, 12})  [inline]
```

Definition at line 261 of file osnma_data.h.

12.404.3.2 Tag() [2/2]

```
Tag::Tag (
            const MACK_message & mack)  [inline], [explicit]
```

Definition at line 284 of file osnma_data.h.

12.404.4 Member Data Documentation

12.404.4.1 ADKD

```
uint8_t Tag::ADKD
```

Definition at line 312 of file osnma_data.h.

12.404.4.2 allowed_adkds

```
std::vector<uint8_t> Tag::allowed_adkds {0, 4, 12}
```

Definition at line 315 of file osnma_data.h.

12.404.4.3 computed_tag

```
uint64_t Tag::computed_tag
```

Definition at line 310 of file osnma_data.h.

12.404.4.4 cop

`uint8_t Tag::cop`
Definition at line 313 of file osnma_data.h.

12.404.4.5 CTR

`uint8_t Tag::CTR`
Definition at line 306 of file osnma_data.h.

12.404.4.6 id_counter

`uint32_t Tag::id_counter [static]`
Definition at line 302 of file osnma_data.h.

12.404.4.7 nav_data

`std::string Tag::nav_data`
Definition at line 316 of file osnma_data.h.

12.404.4.8 nmas

`uint8_t Tag::nmas`
Definition at line 307 of file osnma_data.h.

12.404.4.9 PRN_d

`uint8_t Tag::PRN_d`
Definition at line 311 of file osnma_data.h.

12.404.4.10 PRNa

`uint32_t Tag::PRNa`
Definition at line 305 of file osnma_data.h.

12.404.4.11 received_tag

`uint64_t Tag::received_tag`
Definition at line 309 of file osnma_data.h.

12.404.4.12 skipped

`uint32_t Tag::skipped`
Definition at line 314 of file osnma_data.h.

12.404.4.13 status

`e_verification_status Tag::status`
Definition at line 308 of file osnma_data.h.

12.404.4.14 tag_id

`const uint32_t Tag::tag_id`
Definition at line 301 of file osnma_data.h.

12.404.4.15 TOW

`uint32_t Tag::TOW`
Definition at line 303 of file osnma_data.h.

12.404.4.16 WN

```
uint32_t Tag::WN
```
Definition at line 304 of file osnma_data.h.
The documentation for this class was generated from the following file:

- osnma_data.h

12.405 Tcp_Communication Class Reference

TCP communication class.
`#include <tcp_communication.h>`

Public Member Functions

- int **listen_tcp_connection** (size_t d_port_, size_t d_port_ch0_)
- void **send_receive_tcp_packet_galileo_e1** (boost::array< float, NUM_TX_VARIABLES_GALILEO_E1 > buf, Tcp_Packet_Data *tcp_data_)
- void **send_receive_tcp_packet_gps_l1_ca** (boost::array< float, NUM_TX_VARIABLES_GPS_L1_CA > buf, Tcp_Packet_Data *tcp_data_)
- void **close_tcp_connection** (size_t d_port_)

12.405.1 Detailed Description

TCP communication class.
Definition at line 44 of file tcp_communication.h.
The documentation for this class was generated from the following file:

- tcp_communication.h

12.406 Tcp_Packet_Data Class Reference

Class that implements a TCP data packet.
`#include <tcp_packet_data.h>`

Public Attributes

- float proc_pack_code_error = 0.0
- float proc_pack_carr_error = 0.0
- float proc_pack_carrier_doppler_hz = 0.0

12.406.1 Detailed Description

Class that implements a TCP data packet.
Definition at line 30 of file tcp_packet_data.h.

12.406.2 Member Data Documentation

12.406.2.1 proc_pack_carr_error

```
float Tcp_Packet_Data::proc_pack_carr_error = 0.0
```
Definition at line 36 of file tcp_packet_data.h.

12.406.2.2 proc_pack_carrier_doppler_hz

```
float Tcp_Packet_Data::proc_pack_carrier_doppler_hz = 0.0
```
Definition at line 37 of file tcp_packet_data.h.

12.406.2.3 proc_pack_code_error

```
float Tcp_Packet_Data::proc_pack_code_error = 0.0
```
Definition at line 35 of file tcp_packet_data.h.
The documentation for this class was generated from the following file:

- tcp_packet_data.h

12.407 tcp_t Struct Reference

Public Attributes

- int state
- char saddr [256]
- int port
- struct sockaddr_in addr
- socket_t sock
- int tcon
- unsigned int tact
- unsigned int tdis

12.407.1 Detailed Description

Definition at line 1207 of file rtklib.h.

12.407.2 Member Data Documentation

12.407.2.1 addr

```
struct sockaddr_in tcp_t::addr
```
Definition at line 1212 of file rtklib.h.

12.407.2.2 port

```
int tcp_t::port
```
Definition at line 1211 of file rtklib.h.

12.407.2.3 saddr

```
char tcp_t::saddr[256]
```
Definition at line 1210 of file rtklib.h.

12.407.2.4 sock

```
socket_t tcp_t::sock
```
Definition at line 1213 of file rtklib.h.

12.407.2.5 state

```
int tcp_t::state
```
Definition at line 1209 of file rtklib.h.

12.407.2.6 tact

```
unsigned int tcp_t::tact
```
Definition at line 1215 of file rtklib.h.

12.407.2.7 tcon

`int tcp_t::tcon`
Definition at line 1214 of file rtklib.h.

12.407.2.8 tdis

`unsigned int tcp_t::tdis`
Definition at line 1216 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.408 tcpcli_t Struct Reference

Public Attributes

- tcp_t svr
- int toinact
- int tirecon

12.408.1 Detailed Description

Definition at line 1227 of file rtklib.h.

12.408.2 Member Data Documentation

12.408.2.1 svr

`tcp_t tcpcli_t::svr`
Definition at line 1229 of file rtklib.h.

12.408.2.2 tirecon

`int tcpcli_t::tirecon`
Definition at line 1231 of file rtklib.h.

12.408.2.3 toinact

`int tcpcli_t::toinact`
Definition at line 1230 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.409 TcpCmdInterface Class Reference

Public Member Functions

- void **run_cmd_server** (int tcp_port)
- void **set_msg_queue** (std::shared_ptr< Concurrent_Queue< pmt::pmt_t > > control_queue)
- time_t get_utc_time () const

  gets the UTC time parsed from the last TC command issued
- std::array< float, 3 > get_LLH () const

  gets the Latitude, Longitude and Altitude vector from the last TC command issued
- void **set_pvt** (std::shared_ptr< PvtInterface > PVT_sptr)

12.409.1 Detailed Description

Definition at line 41 of file tcp_cmd_interface.h.

12.409.2 Member Function Documentation

12.409.2.1 get_LLH()

`std::array< float, 3 > TcpCmdInterface::get_LLH () const`
gets the Latitude, Longitude and Altitude vector from the last TC command issued

12.409.2.2 get_utc_time()

`time_t TcpCmdInterface::get_utc_time () const`
gets the UTC time parsed from the last TC command issued
The documentation for this class was generated from the following file:

- tcp_cmd_interface.h

12.410 tcpsvr_t Struct Reference

Public Attributes

- tcp_t svr
- tcp_t cli [MAXCLI]

12.410.1 Detailed Description

Definition at line 1220 of file rtklib.h.

12.410.2 Member Data Documentation

12.410.2.1 cli

`tcp_t tcpsvr_t::cli[MAXCLI]`
Definition at line 1223 of file rtklib.h.

12.410.2.2 svr

`tcp_t tcpsvr_t::svr`
Definition at line 1222 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.411 tec_t Struct Reference

Public Attributes

- gtime_t time
- int ndata [3]
- double rb
- double lats [3]
- double lons [3]
- double hgts [3]
- double * data
- float * rms

12.411.1 Detailed Description

Definition at line 580 of file rtklib.h.

12.411.2 Member Data Documentation

12.411.2.1 data

`double* tec_t::data`
Definition at line 588 of file rtklib.h.

12.411.2.2 hgts

`double tec_t::hgts[3]`
Definition at line 587 of file rtklib.h.

12.411.2.3 lats

`double tec_t::lats[3]`
Definition at line 585 of file rtklib.h.

12.411.2.4 lons

`double tec_t::lons[3]`
Definition at line 586 of file rtklib.h.

12.411.2.5 ndata

`int tec_t::ndata[3]`
Definition at line 583 of file rtklib.h.

12.411.2.6 rb

`double tec_t::rb`
Definition at line 584 of file rtklib.h.

12.411.2.7 rms

`float* tec_t::rms`
Definition at line 589 of file rtklib.h.

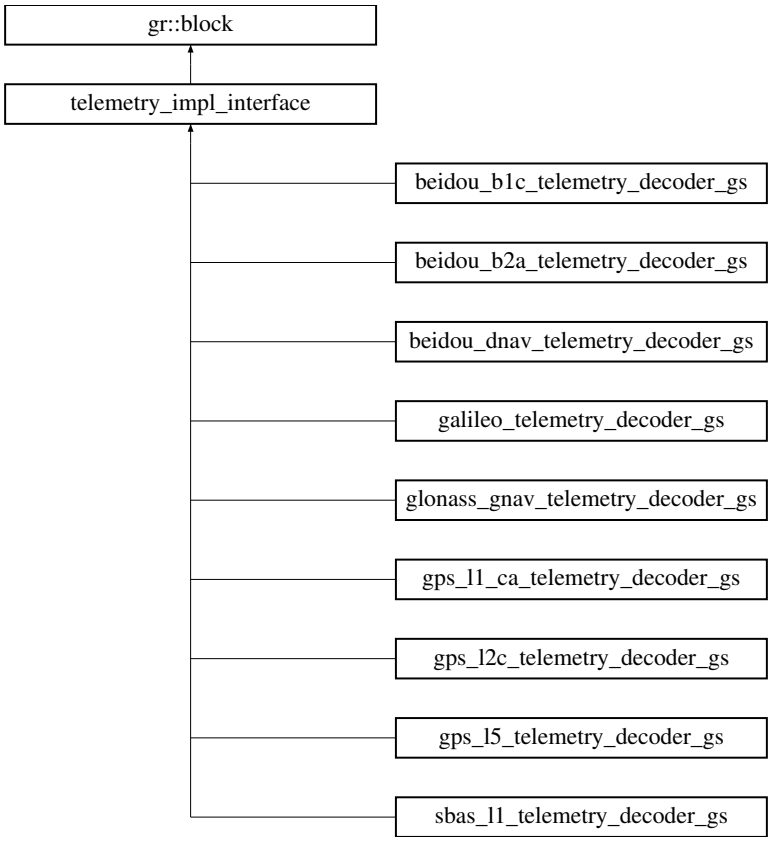
12.411.2.8 time

`gtime_t tec_t::time`
Definition at line 582 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.412 telemetry_impl_interface Class Reference

Common base class for telemetry decoder GNU Radio implementations.
`#include <telemetry_impl_interface.h>`
Inheritance diagram for telemetry_impl_interface:



Public Member Functions

- telemetry_impl_interface (const std::string &name, gr::io_signature::sptr input_signature, gr::io_signature::↩ sptr output_signature)
- virtual void set_satellite (const Gnss_Satellite &satellite)=0
- virtual void set_channel (int channel)=0
- virtual void reset ()=0

Protected Member Functions

- void **configure_basic_outputs** ()
- void **configure_dump_file** (int32_t channel, bool enable_dump, std::string &dump_filename, std::ofstream &dump_file) const
- void **configure_crc_stats_channel** (int32_t channel, bool &dump_crc_stats, std::unique_ptr< Tlm_CRC_Stats > &crc_stats) const

12.412.1 Detailed Description

Common base class for telemetry decoder GNU Radio implementations.
Definition at line 46 of file telemetry_impl_interface.h.

12.412.2 Constructor & Destructor Documentation

12.412.2.1 telemetry_impl_interface()

```
telemetry_impl_interface::telemetry_impl_interface (
            const std::string & name,
            gr::io_signature::sptr input_signature,
            gr::io_signature::sptr output_signature)  [inline]
```

Definition at line 49 of file telemetry_impl_interface.h.

12.412.3 Member Function Documentation

12.412.3.1 reset()

```
virtual void telemetry_impl_interface::reset ()  [pure virtual]
```
Implemented in glonass_gnav_telemetry_decoder_gs.

12.412.3.2 set_channel()

```
virtual void telemetry_impl_interface::set_channel (
            int channel)  [pure virtual]
```
Implemented in beidou_dnav_telemetry_decoder_gs, and gps_l1_ca_telemetry_decoder_gs.

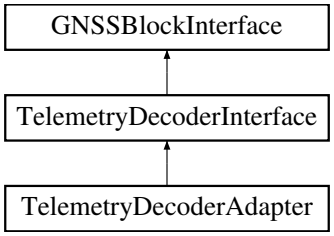
12.412.3.3 set_satellite()

```
virtual void telemetry_impl_interface::set_satellite (
            const Gnss_Satellite & satellite)  [pure virtual]
```
Implemented in beidou_dnav_telemetry_decoder_gs, galileo_telemetry_decoder_gs, glonass_gnav_telemetry_decoder_gs, gps_l1_ca_telemetry_decoder_gs, gps_l2c_telemetry_decoder_gs, gps_l5_telemetry_decoder_gs, and sbas_l1_telemetry_decoder_
The documentation for this class was generated from the following file:

- telemetry_impl_interface.h

12.413 TelemetryDecoderAdapter Class Reference

Base class for Telemetry Decoder adapters.
`#include <telemetry_decoder_adapter.h>`
Inheritance diagram for TelemetryDecoderAdapter:



Public Member Functions

- **TelemetryDecoderAdapter** (const std::string &role, const std::string &implementation, unsigned int in_↩ streams, unsigned int out_streams, telemetry_impl_interface_sptr decoder)
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- void set_satellite (const Gnss_Satellite &satellite) override
- std::string role () override
- std::string implementation () override
- void set_channel (int channel) override
- void reset () override
- size_t item_size () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.413.1 Detailed Description

Base class for Telemetry Decoder adapters.
Definition at line 36 of file telemetry_decoder_adapter.h.

12.413.2 Member Function Documentation

12.413.2.1 connect()

```
void TelemetryDecoderAdapter::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.2 disconnect()

```
void TelemetryDecoderAdapter::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.3 get_left_block()

```
gr::basic_block_sptr TelemetryDecoderAdapter::get_left_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.4 get_right_block()

```
gr::basic_block_sptr TelemetryDecoderAdapter::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.5 implementation()

```
std::string TelemetryDecoderAdapter::implementation ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.6 item_size()

```
size_t TelemetryDecoderAdapter::item_size ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.7 reset()

```
void TelemetryDecoderAdapter::reset ()  [override], [virtual]
```

Implements TelemetryDecoderInterface.

12.413.2.8 role()

```
std::string TelemetryDecoderAdapter::role ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.413.2.9 set_channel()

```
void TelemetryDecoderAdapter::set_channel (
            int channel)  [override], [virtual]
```

Implements TelemetryDecoderInterface.

12.413.2.10 set_satellite()

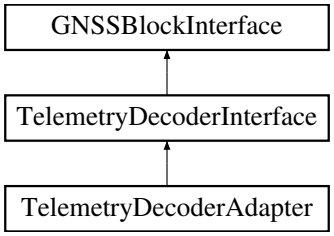
```
void TelemetryDecoderAdapter::set_satellite (
            const Gnss_Satellite & satellite)  [override], [virtual]
```

Implements TelemetryDecoderInterface.
The documentation for this class was generated from the following file:

- telemetry_decoder_adapter.h

12.414 TelemetryDecoderInterface Class Reference

This abstract class represents an interface to a navigation GNSS block.
`#include <telemetry_decoder_interface.h>`
Inheritance diagram for TelemetryDecoderInterface:



Public Member Functions

- virtual void **reset** ()=0
- virtual void **set_satellite** (const Gnss_Satellite &sat)=0
- virtual void **set_channel** (int channel)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.414.1 Detailed Description

This abstract class represents an interface to a navigation GNSS block.
Abstract class for navigation interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 43 of file telemetry_decoder_interface.h.
The documentation for this class was generated from the following file:

- telemetry_decoder_interface.h

12.415 tle_t Struct Reference

Public Attributes

- int n
- int nmax
- tled_t * data

12.415.1 Detailed Description

Definition at line 573 of file rtklib.h.

12.415.2 Member Data Documentation

12.415.2.1 data

```
tled_t* tle_t::data
```
Definition at line 576 of file rtklib.h.

12.415.2.2 n

```
int tle_t::n
```
Definition at line 575 of file rtklib.h.

12.415.2.3 nmax

```
int tle_t::nmax
```
Definition at line 575 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.416 tled_t Struct Reference

Public Attributes

- char name [32]
- char alias [32]
- char satno [16]
- char satclass
- char desig [16]
- gtime_t epoch
- double ndot
- double nddot
- double bstar
- int etype
- int eleno
- double inc
- double OMG
- double ecc
- double omg
- double M
- double n
- int rev

12.416.1 Detailed Description

Definition at line 550 of file rtklib.h.

12.416.2 Member Data Documentation

12.416.2.1 alias

```
char tled_t::alias[32]
```
Definition at line 553 of file rtklib.h.

12.416.2.2 bstar

`double tled_t::bstar`
Definition at line 560 of file rtklib.h.

12.416.2.3 desig

`char tled_t::desig[16]`
Definition at line 556 of file rtklib.h.

12.416.2.4 ecc

`double tled_t::ecc`
Definition at line 565 of file rtklib.h.

12.416.2.5 eleno

`int tled_t::eleno`
Definition at line 562 of file rtklib.h.

12.416.2.6 epoch

`gtime_t tled_t::epoch`
Definition at line 557 of file rtklib.h.

12.416.2.7 etype

`int tled_t::etype`
Definition at line 561 of file rtklib.h.

12.416.2.8 inc

`double tled_t::inc`
Definition at line 563 of file rtklib.h.

12.416.2.9 M

`double tled_t::M`
Definition at line 567 of file rtklib.h.

12.416.2.10 n

`double tled_t::n`
Definition at line 568 of file rtklib.h.

12.416.2.11 name

`char tled_t::name[32]`
Definition at line 552 of file rtklib.h.

12.416.2.12 nddot

`double tled_t::nddot`
Definition at line 559 of file rtklib.h.

12.416.2.13 ndot

`double tled_t::ndot`
Definition at line 558 of file rtklib.h.

12.416.2.14 OMG

`double tled_t::OMG`
Definition at line 564 of file rtklib.h.

12.416.2.15 omg

`double tled_t::omg`
Definition at line 566 of file rtklib.h.

12.416.2.16 rev

`int tled_t::rev`
Definition at line 569 of file rtklib.h.

12.416.2.17 satclass

`char tled_t::satclass`
Definition at line 555 of file rtklib.h.

12.416.2.18 satno

`char tled_t::satno[16]`
Definition at line 554 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.417 Tlm_Conf Class Reference

Public Member Functions

- void **SetFromConfiguration** (const ConfigurationInterface *configuration, const std::string &role)

Public Attributes

- std::string dump_filename
- std::string dump_crc_stats_filename
- bool dump {false}
- bool dump_mat {false}
- bool remove_dat {false}
- bool enable_reed_solomon {false}
- bool dump_crc_stats {false}
- bool enable_navdata_monitor {false}
- bool there_are_e1_channels {false}
- bool there_are_e6_channels {false}
- bool use_ced {false}
- bool tow_to_trk {false}

12.417.1 Detailed Description

Definition at line 30 of file tlm_conf.h.

12.417.2 Member Data Documentation

12.417.2.1 dump

`bool Tlm_Conf::dump {false}`
Definition at line 39 of file tlm_conf.h.

12.417.2.2 dump_crc_stats

```
bool Tlm_Conf::dump_crc_stats {false}
```
Definition at line 43 of file tlm_conf.h.

12.417.2.3 dump_crc_stats_filename

```
std::string Tlm_Conf::dump_crc_stats_filename
```
Definition at line 38 of file tlm_conf.h.

12.417.2.4 dump_filename

```
std::string Tlm_Conf::dump_filename
```
Definition at line 37 of file tlm_conf.h.

12.417.2.5 dump_mat

```
bool Tlm_Conf::dump_mat {false}
```
Definition at line 40 of file tlm_conf.h.

12.417.2.6 enable_navdata_monitor

```
bool Tlm_Conf::enable_navdata_monitor {false}
```
Definition at line 44 of file tlm_conf.h.

12.417.2.7 enable_reed_solomon

```
bool Tlm_Conf::enable_reed_solomon {false}
```
Definition at line 42 of file tlm_conf.h.

12.417.2.8 remove_dat

```
bool Tlm_Conf::remove_dat {false}
```
Definition at line 41 of file tlm_conf.h.

12.417.2.9 there_are_e1_channels

```
bool Tlm_Conf::there_are_e1_channels {false}
```
Definition at line 45 of file tlm_conf.h.

12.417.2.10 there_are_e6_channels

```
bool Tlm_Conf::there_are_e6_channels {false}
```
Definition at line 46 of file tlm_conf.h.

12.417.2.11 tow_to_trk

```
bool Tlm_Conf::tow_to_trk {false}
```
Definition at line 48 of file tlm_conf.h.

12.417.2.12 use_ced

```
bool Tlm_Conf::use_ced {false}
```
Definition at line 47 of file tlm_conf.h.
The documentation for this class was generated from the following file:

- tlm_conf.h

12.418 Tlm_CRC_Stats Class Reference

Class that computes the telemetry CRC statistics.
`#include <tlm_crc_stats.h>`

Public Member Functions

- void initialize (std::string dump_crc_stats_filename)

  Initialize the telemetry CRC statistics.
- bool set_channel (int32_t channel)

  Initialize the channel number and output file.
- void update_CRC_stats (bool CRC)

  Update the CRC statistics.

12.418.1 Detailed Description

Class that computes the telemetry CRC statistics.
Definition at line 32 of file tlm_crc_stats.h.

12.418.2 Member Function Documentation

12.418.2.1 initialize()

```
void Tlm_CRC_Stats::initialize (
            std::string dump_crc_stats_filename)
```

Initialize the telemetry CRC statistics.

12.418.2.2 set_channel()

```
bool Tlm_CRC_Stats::set_channel (
            int32_t channel)
```

Initialize the channel number and output file.

12.418.2.3 update_CRC_stats()

```
void Tlm_CRC_Stats::update_CRC_stats (
            bool CRC)
```

Update the CRC statistics.
The documentation for this class was generated from the following file:

- tlm_crc_stats.h

12.419 TOW_to_trk Class Reference

Public Member Functions

- TOW_to_trk (const std::string &sig, int32_t ch, uint32_t t, uint64_t stamp, int32_t w, uint32_t p)

Public Attributes

- std::string signal
- int32_t channel {0}
- uint32_t tow {0}
- uint64_t sample_stamp {0}
- int32_t wn {0}
- uint32_t prn {0}

12.419.1 Detailed Description

Definition at line 29 of file tow_to_trk.h.

12.419.2 Constructor & Destructor Documentation

12.419.2.1 TOW_to_trk()

```
TOW_to_trk::TOW_to_trk (
            const std::string & sig,
            int32_t ch,
            uint32_t t,
            uint64_t stamp,
            int32_t w,
            uint32_t p)  [inline]
```

Definition at line 35 of file tow_to_trk.h.

12.419.3 Member Data Documentation

12.419.3.1 channel

```
int32_t TOW_to_trk::channel {0}
```

Definition at line 39 of file tow_to_trk.h.

12.419.3.2 prn

```
uint32_t TOW_to_trk::prn {0}
```

Definition at line 43 of file tow_to_trk.h.

12.419.3.3 sample_stamp

```
uint64_t TOW_to_trk::sample_stamp {0}
```

Definition at line 41 of file tow_to_trk.h.

12.419.3.4 signal

```
std::string TOW_to_trk::signal
```

Definition at line 38 of file tow_to_trk.h.

12.419.3.5 tow

```
uint32_t TOW_to_trk::tow {0}
```

Definition at line 40 of file tow_to_trk.h.

12.419.3.6 wn

```
int32_t TOW_to_trk::wn {0}
```

Definition at line 42 of file tow_to_trk.h.
The documentation for this class was generated from the following file:

- tow_to_trk.h

12.420 Monitor_Pvt::TrackedSatelliteInfo Class Reference

One tracked satellite/signal with its azimuth/elevation, whether it was combined with another signal of the same satellite (e.g. Galileo E1+E5a iono-free combination) and whether it was used in this fix. Signals are listed individually (one entry per satellite per signal): a combined satellite appears as two entries, both with combined = true.

```
#include <monitor_pvt.h>
```

Public Member Functions

- template<class Archive>
  void serialize (Archive &ar, const unsigned int version)

Public Attributes

- uint32_t prn {}
- char system {}
- std::string signal
- double azimuth_deg {}
- double elevation_deg {}
- bool combined {}
- bool used {true}
- bool healthy {true}

12.420.1 Detailed Description

One tracked satellite/signal with its azimuth/elevation, whether it was combined with another signal of the same satellite (e.g. Galileo E1+E5a iono-free combination) and whether it was used in this fix. Signals are listed individually (one entry per satellite per signal): a combined satellite appears as two entries, both with combined = true.
Definition at line 45 of file monitor_pvt.h.

12.420.2 Member Function Documentation

12.420.2.1 serialize()

```
template<class Archive>
void Monitor_Pvt::TrackedSatelliteInfo::serialize (
            Archive & ar,
            const unsigned int version)  [inline]
```

Definition at line 62 of file monitor_pvt.h.

12.420.3 Member Data Documentation

12.420.3.1 azimuth_deg

```
double Monitor_Pvt::TrackedSatelliteInfo::azimuth_deg {}
```

Definition at line 51 of file monitor_pvt.h.

12.420.3.2 combined

```
bool Monitor_Pvt::TrackedSatelliteInfo::combined {}
```

Definition at line 53 of file monitor_pvt.h.

12.420.3.3 elevation_deg

```
double Monitor_Pvt::TrackedSatelliteInfo::elevation_deg {}
```

Definition at line 52 of file monitor_pvt.h.

12.420.3.4 healthy

```
bool Monitor_Pvt::TrackedSatelliteInfo::healthy {true}
```

Definition at line 59 of file monitor_pvt.h.

12.420.3.5 prn

```
uint32_t Monitor_Pvt::TrackedSatelliteInfo::prn {}
```

Definition at line 48 of file monitor_pvt.h.

12.420.3.6 signal

```
std::string Monitor_Pvt::TrackedSatelliteInfo::signal
```
Definition at line 50 of file monitor_pvt.h.

12.420.3.7 system

```
char Monitor_Pvt::TrackedSatelliteInfo::system {}
```
Definition at line 49 of file monitor_pvt.h.

12.420.3.8 used

```
bool Monitor_Pvt::TrackedSatelliteInfo::used {true}
```
Definition at line 56 of file monitor_pvt.h.
The documentation for this class was generated from the following file:

- monitor_pvt.h

12.421 Tracking_2nd_DLL_filter Class Reference

This class implements a 2nd order DLL filter for code tracking loop.
`#include <tracking_2nd_DLL_filter.h>`

Public Member Functions

- **Tracking_2nd_DLL_filter** (float pdi_code)
- void set_DLL_BW (float dll_bw_hz)

  Set DLL filter bandwidth [Hz].
- void set_pdi (float pdi_code)

  Set Summation interval for code [s].
- void initialize ()

  Start tracking with acquisition information.
- float get_code_nco (float DLL_discriminator)

  Numerically controlled oscillator.

12.421.1 Detailed Description

This class implements a 2nd order DLL filter for code tracking loop.
The algorithm is described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S. H. Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
Definition at line 40 of file tracking_2nd_DLL_filter.h.

12.421.2 Member Function Documentation

12.421.2.1 get_code_nco()

```
float Tracking_2nd_DLL_filter::get_code_nco (
            float DLL_discriminator)
```
Numerically controlled oscillator.

12.421.2.2 initialize()

```
void Tracking_2nd_DLL_filter::initialize ()
```
Start tracking with acquisition information.

12.421.2.3 set_DLL_BW()

```
void Tracking_2nd_DLL_filter::set_DLL_BW (
            float dll_bw_hz)
```

Set DLL filter bandwidth [Hz].

12.421.2.4 set_pdi()

```
void Tracking_2nd_DLL_filter::set_pdi (
            float pdi_code)
```

Set Summation interval for code [s].
The documentation for this class was generated from the following file:

- tracking_2nd_DLL_filter.h

12.422 Tracking_2nd_PLL_filter Class Reference

This class implements a 2nd order PLL filter for carrier tracking loop.
`#include <tracking_2nd_PLL_filter.h>`

Public Member Functions

- **Tracking_2nd_PLL_filter** (float pdi_carr)
- void set_PLL_BW (float pll_bw_hz)

  Set PLL loop bandwidth [Hz].
- void set_pdi (float pdi_carr)

  Set Summation interval for code [s].
- void **initialize** ()
- float **get_carrier_nco** (float PLL_discriminator)

12.422.1 Detailed Description

This class implements a 2nd order PLL filter for carrier tracking loop.
The algorithm is described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S. H. Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
Definition at line 39 of file tracking_2nd_PLL_filter.h.

12.422.2 Member Function Documentation

12.422.2.1 set_pdi()

```
void Tracking_2nd_PLL_filter::set_pdi (
            float pdi_carr)
```

Set Summation interval for code [s].

12.422.2.2 set_PLL_BW()

```
void Tracking_2nd_PLL_filter::set_PLL_BW (
            float pll_bw_hz)
```

Set PLL loop bandwidth [Hz].
The documentation for this class was generated from the following file:

- tracking_2nd_PLL_filter.h

12.423 Tracking_FLL_PLL_filter Class Reference

This class implements a hybrid FLL and PLL filter for tracking carrier loop.
`#include <tracking_FLL_PLL_filter.h>`

Public Member Functions

- void **set_params** (float fll_bw_hz, float pll_bw_hz, int order)
- void **initialize** (float d_acq_carrier_doppler_hz)
- float **get_carrier_error** (float FLL_discriminator, float PLL_discriminator, float correlation_time_s)

12.423.1 Detailed Description

This class implements a hybrid FLL and PLL filter for tracking carrier loop.
Definition at line 29 of file tracking_FLL_PLL_filter.h.
The documentation for this class was generated from the following file:

- tracking_FLL_PLL_filter.h

12.424 Tracking_loop_filter Class Reference

This class implements a generic 1st, 2nd or 3rd order loop filter.
`#include <tracking_loop_filter.h>`

Public Member Functions

- **Tracking_loop_filter** (float update_interval, float noise_bandwidth, int loop_order=2, bool include_last_integrator=false)
- Tracking_loop_filter (Tracking_loop_filter &&)=default
  Move operator.
- Tracking_loop_filter & operator= (Tracking_loop_filter &&)=default
  Move assignment operator.
- float **get_noise_bandwidth** () const
- float **get_update_interval** () const
- bool **get_include_last_integrator** () const
- int **get_order** () const
- void **set_noise_bandwidth** (float noise_bandwidth)
- void **set_update_interval** (float update_interval)
- void **set_include_last_integrator** (bool include_last_integrator)
- void **set_order** (int loop_order)
- void **initialize** (float initial_output=0.0)
- float **apply** (float current_input)

12.424.1 Detailed Description

This class implements a generic 1st, 2nd or 3rd order loop filter.
Definition at line 35 of file tracking_loop_filter.h.

12.424.2 Constructor & Destructor Documentation

12.424.2.1 Tracking_loop_filter()

```
Tracking_loop_filter::Tracking_loop_filter (
            Tracking_loop_filter && )  [default]
```

Move operator.

12.424.3 Member Function Documentation

12.424.3.1 operator=()

```
Tracking_loop_filter & Tracking_loop_filter::operator= (
            Tracking_loop_filter && )  [default]
```

Move assignment operator.
The documentation for this class was generated from the following file:

- tracking_loop_filter.h

12.425 TrackingCmd Class Reference

Public Attributes

- bool enable_carrier_nco_cmd = false
- bool enable_code_nco_cmd = false
- double code_freq_chips = 0.0
- double carrier_freq_hz = 0.0
- double carrier_freq_rate_hz_s = 0.0
- uint64_t sample_counter = 0UL

12.425.1 Detailed Description

Definition at line 28 of file trackingcmd.h.

12.425.2 Member Data Documentation

12.425.2.1 carrier_freq_hz

```
double TrackingCmd::carrier_freq_hz = 0.0
```

Definition at line 36 of file trackingcmd.h.

12.425.2.2 carrier_freq_rate_hz_s

```
double TrackingCmd::carrier_freq_rate_hz_s = 0.0
```

Definition at line 37 of file trackingcmd.h.

12.425.2.3 code_freq_chips

```
double TrackingCmd::code_freq_chips = 0.0
```

Definition at line 35 of file trackingcmd.h.

12.425.2.4 enable_carrier_nco_cmd

```
bool TrackingCmd::enable_carrier_nco_cmd = false
```

Definition at line 33 of file trackingcmd.h.

12.425.2.5 enable_code_nco_cmd

```
bool TrackingCmd::enable_code_nco_cmd = false
```

Definition at line 34 of file trackingcmd.h.

12.425.2.6 sample_counter

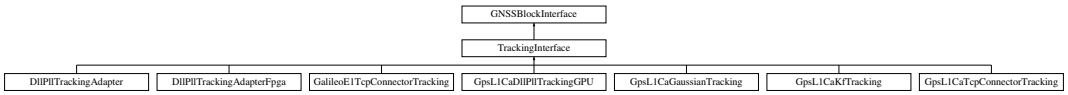
```
uint64_t TrackingCmd::sample_counter = 0UL
```

Definition at line 38 of file trackingcmd.h.
The documentation for this class was generated from the following file:

- trackingcmd.h

12.426 TrackingInterface Class Reference

This abstract class represents an interface to a tracking block.
`#include <tracking_interface.h>`
Inheritance diagram for TrackingInterface:



Public Member Functions

- virtual void start_tracking ()=0
- virtual void stop_tracking ()=0
- virtual void set_gnss_synchro (Gnss_Synchro *gnss_synchro)=0
- virtual void set_channel (unsigned int channel)=0

Public Member Functions inherited from GNSSBlockInterface

- virtual std::string role ()=0
- virtual std::string implementation ()=0
- virtual size_t item_size ()=0
- virtual void connect (gr::top_block_sptr top_block)=0
- virtual void disconnect (gr::top_block_sptr top_block)=0
- virtual gr::basic_block_sptr get_left_block ()=0
- virtual gr::basic_block_sptr get_right_block ()=0
- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

12.426.1 Detailed Description

This abstract class represents an interface to a tracking block.
Abstract class for tracking interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.
Definition at line 47 of file tracking_interface.h.

12.426.2 Member Function Documentation

12.426.2.1 set_channel()

```
virtual void TrackingInterface::set_channel (
            unsigned int channel)  [pure virtual]
```

Implemented in DllPllTrackingAdapter, GalileoE1TcpConnectorTracking, GpsL1CaDllPllTrackingGPU, GpsL1CaGaussianTracking, GpsL1CaKfTracking, and GpsL1CaTcpConnectorTracking.

12.426.2.2 set_gnss_synchro()

```
virtual void TrackingInterface::set_gnss_synchro (
            Gnss_Synchro * gnss_synchro)  [pure virtual]
```

Implemented in DllPllTrackingAdapter, GalileoE1TcpConnectorTracking, GpsL1CaDllPllTrackingGPU, GpsL1CaGaussianTracking, GpsL1CaKfTracking, and GpsL1CaTcpConnectorTracking.

12.426.2.3 start_tracking()

```
virtual void TrackingInterface::start_tracking ()  [pure virtual]
```

Implemented in DllPllTrackingAdapter.

12.426.2.4 stop_tracking()

```
virtual void TrackingInterface::stop_tracking ()  [pure virtual]
```

Implemented in DllPllTrackingAdapter, GalileoE1TcpConnectorTracking, GpsL1CaDllPllTrackingGPU, GpsL1CaGaussianTracking, GpsL1CaKfTracking, and GpsL1CaTcpConnectorTracking.

The documentation for this class was generated from the following file:

- tracking_interface.h

12.427 trop_t Struct Reference

Public Attributes

- gtime_t time
- double trp [3]
- float std [3]

12.427.1 Detailed Description

Definition at line 768 of file rtklib.h.

12.427.2 Member Data Documentation

12.427.2.1 std

```
float trop_t::std[3]
```

Definition at line 772 of file rtklib.h.

12.427.2.2 time

```
gtime_t trop_t::time
```

Definition at line 770 of file rtklib.h.

12.427.2.3 trp

```
double trop_t::trp[3]
```

Definition at line 771 of file rtklib.h.

The documentation for this struct was generated from the following file:

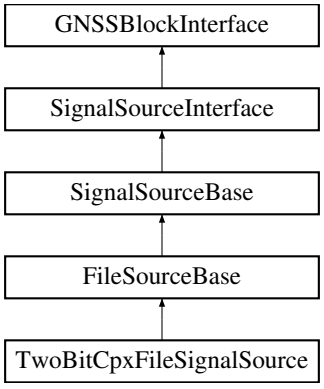
- rtklib.h

12.428 TwoBitCpxFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

```
#include <two_bit_cpx_file_signal_source.h>
```

Inheritance diagram for TwoBitCpxFileSignalSource:



Public Member Functions

- **TwoBitCpxFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- double packetsPerSample () const override

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

  For complex source chains, the size of the file item may not be the same as the size of the.
- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.428.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 42 of file two_bit_cpx_file_signal_source.h.

12.428.2 Member Function Documentation

12.428.2.1 create_file_source_hook()

```
void TwoBitCpxFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.428.2.2 itemTypeToSize()

```
std::tuple< size_t, bool > TwoBitCpxFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```

Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.428.2.3 packetsPerSample()

```
double TwoBitCpxFileSignalSource::packetsPerSample () const  [override], [protected], [virtual]
```

The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented from FileSourceBase.

12.428.2.4 pre_connect_hook()

```
void TwoBitCpxFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.428.2.5 pre_disconnect_hook()

```
void TwoBitCpxFileSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.428.2.6 source()

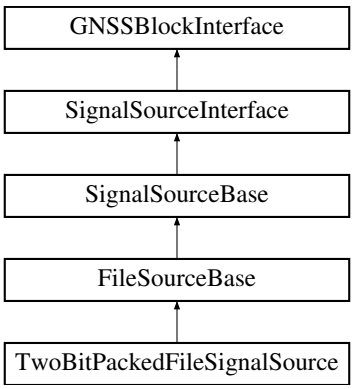
```
gnss_shared_ptr< gr::block > TwoBitCpxFileSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- two_bit_cpx_file_signal_source.h

12.429 TwoBitPackedFileSignalSource Class Reference

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
`#include <two_bit_packed_file_signal_source.h>`
Inheritance diagram for TwoBitPackedFileSignalSource:



Public Member Functions

- **TwoBitPackedFileSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue)

Public Member Functions inherited from FileSourceBase

- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override

- gr::basic_block_sptr get_right_block () override
- std::string filename () const

  The file to read.
- std::string item_type () const

  The item type.
- size_t item_size () override

  The configured size of each item.
- virtual size_t **item_size** () const
- bool repeat () const

  Whether to repeat reading after end-of-file.
- int64_t sampling_frequency () const

  The sampling frequency of the source file.
- uint64_t samples () const

  The number of samples in the file.

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Protected Member Functions

- std::tuple< size_t, bool > itemTypeToSize () override

  Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
- double packetsPerSample () const override

  The number of (possibly unpacked) samples in a (raw) file sample (default=1).
- gnss_shared_ptr< gr::block > source () const override

  Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
- void create_file_source_hook () override
- void pre_connect_hook (gr::top_block_sptr top_block) override
- void pre_disconnect_hook (gr::top_block_sptr top_block) override

Protected Member Functions inherited from FileSourceBase

- FileSourceBase (ConfigurationInterface const *configuration, std::string const &role, std::string impl, Concurrent_Queue< pmt::pmt_t > *queue, std::string default_item_type="short")

  Constructor.
- void init ()

  Perform post-construction initialization.
- virtual uint64_t samplesToSkip () const

  Compute the number of samples to skip.
- uint64_t computeSamplesInFile () const

  Compute the number of samples in the file.
- virtual size_t source_item_size () const

For complex source chains, the size of the file item may not be the same as the size of the.

- bool **is_complex** () const
- gnss_shared_ptr< gr::block > **file_source** () const
- gnss_shared_ptr< gr::block > **valve** () const
- gnss_shared_ptr< gr::block > **throttle** () const
- gnss_shared_ptr< gr::block > **sink** () const
- SensorDataSource::sptr **sensor_data_source** () const
- gr::blocks::file_source::sptr **create_file_source** ()
- gr::blocks::throttle::sptr **create_throttle** ()
- gnss_shared_ptr< gr::block > **create_valve** ()
- gr::blocks::file_sink::sptr **create_sink** ()
- SensorDataSource::sptr **create_sensor_data_source** ()
- virtual void **create_throttle_hook** ()
- virtual void **create_valve_hook** ()
- virtual void **create_sink_hook** ()
- virtual void **post_connect_hook** (gr::top_block_sptr top_block)
- virtual void **post_disconnect_hook** (gr::top_block_sptr top_block)

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.429.1 Detailed Description

Class that reads signals samples from a file and adapts it to a SignalSourceInterface.
Definition at line 44 of file two_bit_packed_file_signal_source.h.

12.429.2 Member Function Documentation

12.429.2.1 create_file_source_hook()

```
void TwoBitPackedFileSignalSource::create_file_source_hook ()  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.429.2.2 itemTypeToSize()

```
std::tuple< size_t, bool > TwoBitPackedFileSignalSource::itemTypeToSize ()  [override], [protected],
[virtual]
```

Compute the item size, from the item_type(). Subclasses may constrain types that don't make.
Reimplemented from FileSourceBase.

12.429.2.3 packetsPerSample()

```
double TwoBitPackedFileSignalSource::packetsPerSample () const  [override], [protected], [virtual]
```

The number of (possibly unpacked) samples in a (raw) file sample (default=1).
Reimplemented from FileSourceBase.

12.429.2.4 pre_connect_hook()

```
void TwoBitPackedFileSignalSource::pre_connect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.429.2.5 pre_disconnect_hook()

```
void TwoBitPackedFileSignalSource::pre_disconnect_hook (
            gr::top_block_sptr top_block)  [override], [protected], [virtual]
```

Reimplemented from FileSourceBase.

12.429.2.6 source()

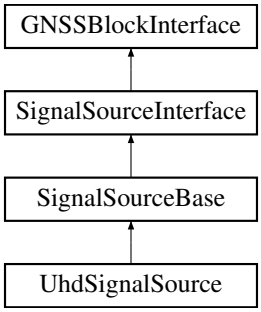
```
gnss_shared_ptr< gr::block > TwoBitPackedFileSignalSource::source () const  [override], [protected],
[virtual]
```

Abstracted front-end source. Sub-classes may override if they create specialized chains to decode source files into a usable format.
Reimplemented from FileSourceBase.
The documentation for this class was generated from the following file:

- two_bit_packed_file_signal_source.h

12.430 UhdSignalSource Class Reference

This class reads samples from a UHD device (see http://code.ettus.com/redmine/ettus/↩ projects/uhd/wiki).
`#include <uhd_signal_source.h>`
Inheritance diagram for UhdSignalSource:



Public Member Functions

- **UhdSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- void connect (gr::top_block_sptr top_block) override
- void disconnect (gr::top_block_sptr top_block) override
- gr::basic_block_sptr get_left_block () override
- gr::basic_block_sptr get_right_block () override
- gr::basic_block_sptr get_right_block (int RF_channel) override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_error=false)

  utility for decoding passed ".item_type" values

12.430.1 Detailed Description

This class reads samples from a UHD device (see http://code.ettus.com/redmine/ettus/projects/uhd/wiki).
Definition at line 41 of file uhd_signal_source.h.

12.430.2 Member Function Documentation

12.430.2.1 connect()

```
void UhdSignalSource::connect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.430.2.2 disconnect()

```
void UhdSignalSource::disconnect (
            gr::top_block_sptr top_block)  [override], [virtual]
```

Implements GNSSBlockInterface.

12.430.2.3 get_left_block()

```
gr::basic_block_sptr UhdSignalSource::get_left_block ()  [override], [virtual]
```

Reimplemented from SignalSourceBase.

12.430.2.4 get_right_block() [1/2]

```
gr::basic_block_sptr UhdSignalSource::get_right_block ()  [override], [virtual]
```

Implements GNSSBlockInterface.

12.430.2.5 get_right_block() [2/2]

```
gr::basic_block_sptr UhdSignalSource::get_right_block (
            int RF_channel)  [override], [virtual]
```

Reimplemented from GNSSBlockInterface.

12.430.2.6 item_size()

```
size_t UhdSignalSource::item_size ()  [inline], [override], [virtual]
```

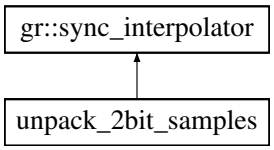
Implements GNSSBlockInterface.
Definition at line 50 of file uhd_signal_source.h.
The documentation for this class was generated from the following file:

- uhd_signal_source.h

12.431 unpack_2bit_samples Class Reference

This class takes 2 bit samples that have been packed into bytes or shorts as input and generates a byte for each sample. It generates eight times as much data as is input (every two bits become 16 bits).

`#include <unpack_2bit_samples.h>`
Inheritance diagram for unpack_2bit_samples:



Public Member Functions

- **unpack_2bit_samples** (bool big_endian_bytes, size_t item_size, bool big_endian_items, bool reverse_↩ interleaving)
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_2bit_samples_sptr **make_unpack_2bit_samples_sptr** (bool big_endian_bytes, size_t item_size, bool big_endian_items, bool reverse_interleaving)

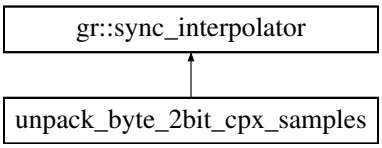
12.431.1 Detailed Description

This class takes 2 bit samples that have been packed into bytes or shorts as input and generates a byte for each sample. It generates eight times as much data as is input (every two bits become 16 bits).
Definition at line 84 of file unpack_2bit_samples.h.
The documentation for this class was generated from the following file:

- unpack_2bit_samples.h

12.432 unpack_byte_2bit_cpx_samples Class Reference

This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples.
`#include <unpack_byte_2bit_cpx_samples.h>`
Inheritance diagram for unpack_byte_2bit_cpx_samples:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_byte_2bit_cpx_samples_sptr **make_unpack_byte_2bit_cpx_samples_sptr** ()

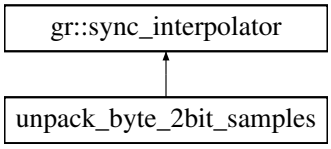
12.432.1 Detailed Description

This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples.
Definition at line 44 of file unpack_byte_2bit_cpx_samples.h.
The documentation for this class was generated from the following file:

- unpack_byte_2bit_cpx_samples.h

12.433 unpack_byte_2bit_samples Class Reference

This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.
`#include <unpack_byte_2bit_samples.h>`
Inheritance diagram for unpack_byte_2bit_samples:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_byte_2bit_samples_sptr **make_unpack_byte_2bit_samples_sptr** ()

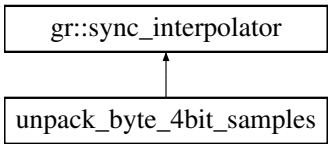
12.433.1 Detailed Description

This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.
Definition at line 40 of file unpack_byte_2bit_samples.h.
The documentation for this class was generated from the following file:

- unpack_byte_2bit_samples.h

12.434 unpack_byte_4bit_samples Class Reference

This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples.
`#include <unpack_byte_4bit_samples.h>`
Inheritance diagram for unpack_byte_4bit_samples:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_byte_4bit_samples_sptr **make_unpack_byte_4bit_samples_sptr** ()

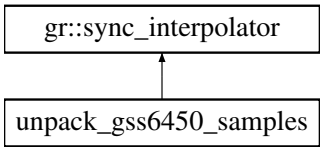
12.434.1 Detailed Description

This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples.
Definition at line 42 of file unpack_byte_4bit_samples.h.
The documentation for this class was generated from the following file:

- unpack_byte_4bit_samples.h

12.435 unpack_gss6450_samples Class Reference

Inheritance diagram for unpack_gss6450_samples:



Public Member Functions

- **unpack_gss6450_samples** (int adc_nbit, bool gss6425_compatible=false)
- void **decode_word** (uint32_t input_uint32, gr_complex ∗out) const
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_gss6450_samples_sptr **make_unpack_gss6450_samples** (int adc_nbit_, bool gss6425_↩ compatible=false)

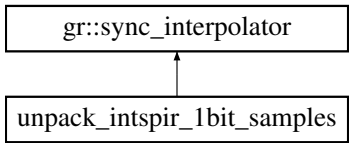
12.435.1 Detailed Description

Definition at line 39 of file unpack_gss6450_samples.h.
The documentation for this class was generated from the following file:

- unpack_gss6450_samples.h

12.436 unpack_intspir_1bit_samples Class Reference

This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.
`#include <unpack_intspir_1bit_samples.h>`
Inheritance diagram for unpack_intspir_1bit_samples:



Public Member Functions

- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_intspir_1bit_samples_sptr **make_unpack_intspir_1bit_samples_sptr** ()

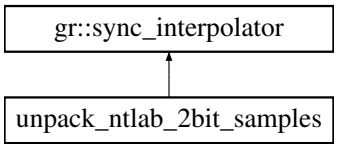
12.436.1 Detailed Description

This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.
Definition at line 41 of file unpack_intspir_1bit_samples.h.
The documentation for this class was generated from the following file:

- unpack_intspir_1bit_samples.h

12.437 unpack_ntlab_2bit_samples Class Reference

This class implements conversion between byte packet multichannel samples to 2bit samples 1 byte = 4 2bit samples.
`#include <unpack_ntlab_2bit_samples.h>`
Inheritance diagram for unpack_ntlab_2bit_samples:



Public Member Functions

- **unpack_ntlab_2bit_samples** (size_t item_size, int nchannels)
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_ntlab_2bit_samples_sptr **make_unpack_ntlab_2bit_samples_sptr** (size_t item_size, int nchannels)

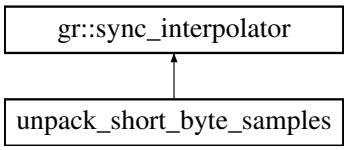
12.437.1 Detailed Description

This class implements conversion between byte packet multichannel samples to 2bit samples 1 byte = 4 2bit samples.
Unpack each of the four 2-bit samples in the byte 'b' into four real-valued outputs.
The NTLAB format encodes samples as sign+magnitude pairs in each byte: bits 7-6 = [M0 S0] -> sample 0 bits 5-4 = [M1 S1] -> sample 1 bits 3-2 = [M2 S2] -> sample 2 bits 1-0 = [M3 S3] -> sample 3
M = magnitude bit (1->|sample|=3, 0->|sample|=1) S = sign bit (1->positive, 0->negative)
Definition at line 56 of file unpack_ntlab_2bit_samples.h.
The documentation for this class was generated from the following file:

- unpack_ntlab_2bit_samples.h

12.438 unpack_short_byte_samples Class Reference

This class implements conversion between short packet samples to byte samples 1 short = 2 byte samples.
`#include <unpack_short_byte_samples.h>`
Inheritance diagram for unpack_short_byte_samples:



Public Member Functions

- void **forecast** (int noutput_items, gr_vector_int &ninput_items_required)
- int **work** (int noutput_items, gr_vector_const_void_star &input_items, gr_vector_void_star &output_items)

Friends

- unpack_short_byte_samples_sptr **make_unpack_short_byte_samples_sptr** ()

12.438.1 Detailed Description

This class implements conversion between short packet samples to byte samples 1 short = 2 byte samples.
Definition at line 41 of file unpack_short_byte_samples.h.
The documentation for this class was generated from the following file:

- unpack_short_byte_samples.h

12.439 UnscentedFilter Class Reference

Public Member Functions

- **UnscentedFilter** (int nx)
- **UnscentedFilter** (const arma::vec &x_pred_0, const arma::mat &P_x_pred_0)
- void **initialize** (const arma::mat &x_pred_0, const arma::mat &P_x_pred_0)
- void **predict_sequential** (const arma::vec &x_post, const arma::mat &P_x_post, ModelFunction *transition↩ _fcn, const arma::mat &noise_covariance)
- void **update_sequential** (const arma::vec &z_upd, const arma::vec &x_pred, const arma::mat &P_x_pred, ModelFunction *measurement_fcn, const arma::mat &noise_covariance)
- arma::mat **get_x_pred** () const
- arma::mat **get_P_x_pred** () const
- arma::mat **get_x_est** () const
- arma::mat **get_P_x_est** () const

12.439.1 Detailed Description

Definition at line 83 of file nonlinear_tracking.h.
The documentation for this class was generated from the following file:

- nonlinear_tracking.h

12.440 url_t Struct Reference

Public Attributes

- char type [32]
- char path [1024]
- char dir [1024]
- double tint

12.440.1 Detailed Description

Definition at line 947 of file rtklib.h.

12.440.2 Member Data Documentation

12.440.2.1 dir

```
char url_t::dir[1024]
```

Definition at line 951 of file rtklib.h.

12.440.2.2 path

```
char url_t::path[1024]
```

Definition at line 950 of file rtklib.h.

12.440.2.3 tint

```
double url_t::tint
```

Definition at line 952 of file rtklib.h.

12.440.2.4 type

```
char url_t::type[32]
```
Definition at line 949 of file rtklib.h.
The documentation for this struct was generated from the following file:

- rtklib.h

12.441 v27_decision_t Struct Reference

Public Attributes

- unsigned int w [2]

12.441.1 Detailed Description

Definition at line 38 of file fec.h.

12.441.2 Member Data Documentation

12.441.2.1 w

```
unsigned int v27_decision_t::w[2]
```
Definition at line 40 of file fec.h.
The documentation for this struct was generated from the following file:

- fec.h

12.442 v27_poly_t Struct Reference

Public Attributes

- unsigned char c0 [32]
- unsigned char c1 [32]

12.442.1 Detailed Description

Definition at line 32 of file fec.h.

12.442.2 Member Data Documentation

12.442.2.1 c0

```
unsigned char v27_poly_t::c0[32]
```
Definition at line 34 of file fec.h.

12.442.2.2 c1

```
unsigned char v27_poly_t::c1[32]
```
Definition at line 35 of file fec.h.
The documentation for this struct was generated from the following file:

- fec.h

12.443 v27_t Struct Reference

Public Attributes

- unsigned int metrics1 [64]
- unsigned int metrics2 [64]

- unsigned int * old_metrics
- unsigned int * new_metrics
- const v27_poly_t * poly
- v27_decision_t * decisions
- unsigned int decisions_index
- unsigned int decisions_count

12.443.1 Detailed Description

Definition at line 45 of file fec.h.

12.443.2 Member Data Documentation

12.443.2.1 decisions

```
v27_decision_t* v27_t::decisions
```

Definition at line 52 of file fec.h.

12.443.2.2 decisions_count

```
unsigned int v27_t::decisions_count
```

Definition at line 54 of file fec.h.

12.443.2.3 decisions_index

```
unsigned int v27_t::decisions_index
```

Definition at line 53 of file fec.h.

12.443.2.4 metrics1

```
unsigned int v27_t::metrics1[64]
```

Definition at line 47 of file fec.h.

12.443.2.5 metrics2

```
unsigned int v27_t::metrics2[64]
```

Definition at line 48 of file fec.h.

12.443.2.6 new_metrics

```
unsigned int * v27_t::new_metrics
```

Definition at line 50 of file fec.h.

12.443.2.7 old_metrics

```
unsigned int* v27_t::old_metrics
```

Definition at line 50 of file fec.h.

12.443.2.8 poly

```
const v27_poly_t* v27_t::poly
```

Definition at line 51 of file fec.h.

The documentation for this struct was generated from the following file:

- fec.h

12.444 Viterbi_Decoder Class Reference

Class that implements a Viterbi decoder.

```
#include <viterbi_decoder.h>
```

Public Member Functions

- Viterbi_Decoder (int32_t KK, int32_t nn, int32_t LL, const std::array< int32_t, 2 > &g)

  Constructor of a Viterbi decoder.
- void decode (std::vector< int32_t > &output_u_int, const std::vector< float > &input_c)

  Uses the Viterbi algorithm to perform soft-decision decoding of a convolutional code.
- void reset ()

  Reset internal status.

12.444.1 Detailed Description

Class that implements a Viterbi decoder.
Definition at line 35 of file viterbi_decoder.h.

12.444.2 Constructor & Destructor Documentation

12.444.2.1 Viterbi_Decoder()

```
Viterbi_Decoder::Viterbi_Decoder (
            int32_t KK,
            int32_t nn,
            int32_t LL,
            const std::array< int32_t, 2 > & g)
```

Constructor of a Viterbi decoder.

Parameters

in	*KK*	Constraint length
in	*nn*	Coding rate 1/n
in	*LL*	Data length
in	*g*	Polynomial G1 and G2

12.444.3 Member Function Documentation

12.444.3.1 decode()

```
void Viterbi_Decoder::decode (
            std::vector< int32_t > & output_u_int,
            const std::vector< float > & input_c)
```

Uses the Viterbi algorithm to perform soft-decision decoding of a convolutional code.

Parameters

out	*output_u_int*	Hard decisions on the decoded data bits.
in	*input_c*	Soft input metrics.

12.444.3.2 reset()

```
void Viterbi_Decoder::reset ()
```

Reset internal status.
The documentation for this class was generated from the following file:

- viterbi_decoder.h

12.445 Viterbi_Decoder_Sbas Class Reference

Class that implements a Viterbi decoder.
`#include <viterbi_decoder_sbas.h>`

Public Member Functions

- **Viterbi_Decoder_Sbas** (const int g_encoder[ ], int KK, int nn)
- void **reset** ()
- float decode_block (const double input_c[ ], int ∗output_u_int, int LL)

  Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.
- float **decode_continuous** (const double sym[ ], int traceback_depth, int bits[ ], int nbits_requested, int &nbits_decoded)

12.445.1 Detailed Description

Class that implements a Viterbi decoder.
Definition at line 34 of file viterbi_decoder_sbas.h.

12.445.2 Member Function Documentation

12.445.2.1 decode_block()

```
float Viterbi_Decoder_Sbas::decode_block (
            const double input_c[],
            int * output_u_int,
            int LL)
```

Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.

Parameters

in	*input_c[]*	The received signal in LLR-form. For BPSK, must be in form r = 2∗a∗y/(sigma^2).
in	*LL*	The number of data bits to be decoded (does not include the mm zero-tail-bits)

Returns

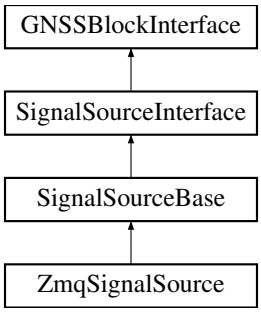
output_u_int[] Hard decisions on the data bits (without the mm zero-tail-bits)

The documentation for this class was generated from the following file:

- viterbi_decoder_sbas.h

12.446 ZmqSignalSource Class Reference

This class supports the following properties:
`#include <zmq_signal_source.h>`
Inheritance diagram for ZmqSignalSource:



Public Member Functions

- **ZmqSignalSource** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_↩ stream, unsigned int out_stream, Concurrent_Queue< pmt::pmt_t > *queue)
- size_t item_size () override
- auto connect (gr::top_block_sptr top_block) -> void override
- auto disconnect (gr::top_block_sptr top_block) -> void override
- auto get_right_block () -> gr::basic_block_sptr override

Public Member Functions inherited from SignalSourceBase

- std::string role () final
- std::string implementation () final
- size_t getRfChannels () const override
- gr::basic_block_sptr get_left_block () override

Public Member Functions inherited from GNSSBlockInterface

- virtual gr::basic_block_sptr get_left_block (int RF_channel)
- virtual gr::basic_block_sptr get_right_block (int RF_channel)
- virtual void start ()

  Start the flow of samples if needed.

Additional Inherited Members

Protected Member Functions inherited from SignalSourceBase

- SignalSourceBase (ConfigurationInterface const *configuration, std::string role, std::string impl)

  Constructor.
- size_t decode_item_type (std::string const &item_type, bool *is_interleaved=nullptr, bool throw_on_↩ error=false)

  utility for decoding passed ".item_type" values

12.446.1 Detailed Description

This class supports the following properties:
.endpoint - the ZMQ endpoint to be connected to .vlen - vector length of the input items (default 1, one item) this must match the size of the publisher! .pass_tags - boolean flag if tags should be propagated (default false) .timeout_ms - receive timeout, in milliseconds (default 100) .hwm - ZMQ high water mark (default -1, ZMQ default)
.item_type - data type of the samples (default "gr_complex")
(probably should be abstracted to the base class)
.dump - whether to archive input data .dump_filename - if dumping, path to file for output
Definition at line 51 of file zmq_signal_source.h.

12.446.2 Member Function Documentation

12.446.2.1 connect()

```
auto ZmqSignalSource::connect (
            gr::top_block_sptr top_block) -> void  [override], [virtual]
```

Implements GNSSBlockInterface.

12.446.2.2 disconnect()

```
auto ZmqSignalSource::disconnect (
            gr::top_block_sptr top_block) -> void  [override], [virtual]
```

Implements GNSSBlockInterface.

12.446.2.3 get_right_block()

```
auto ZmqSignalSource::get_right_block () -> gr::basic_block_sptr  [override], [virtual]
```

Implements GNSSBlockInterface.

12.446.2.4 item_size()

```
size_t ZmqSignalSource::item_size ()  [override], [virtual]
```

Implements GNSSBlockInterface.

The documentation for this class was generated from the following file:

- zmq_signal_source.h

Chapter 13

File Documentation

13.1 pcps_acquisition_adapter.h File Reference

Adapts a PCPS acquisition block to an AcquisitionInterface.

```
#include "acq_conf.h"
#include "channel_fsm.h"
#include "complex_byte_to_float_x2.h"
#include "gnss_synchro.h"
#include "pcps_acquisition.h"
#include "signal_flag.h"
#include <gnuradio/blocks/float_to_complex.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
```

Classes

- class PcpsAcquisitionAdapter

  This class adapts a PCPS acquisition block to an AcquisitionInterface.

13.1.1 Detailed Description

Adapts a PCPS acquisition block to an AcquisitionInterface.

Authors

- Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_acquisition_adapter.h.

13.2 pcps_acquisition_adapter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_acquisition_adapter.h
00003  * \brief Adapts a PCPS acquisition block to an AcquisitionInterface
00004  * \authors <ul>
00005  *          <li> Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com
00006  *          </ul>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
```

```
00017  */
00018
00019 #ifndef GNSS_SDR_PCPS_ACQUISITION_ADAPTER_H
00020 #define GNSS_SDR_PCPS_ACQUISITION_ADAPTER_H
00021
00022 #include "acq_conf.h"
00023 #include "channel_fsm.h"
00024 #include "complex_byte_to_float_x2.h"
00025 #include "gnss_synchro.h"
00026 #include "pcps_acquisition.h"
00027 #include "signal_flag.h"
00028 #include <gnuradio/blocks/float_to_complex.h>
00029 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>
00030
00031 /** \addtogroup Acquisition
00032  * Classes for GNSS signal acquisition
00033  * \{ */
00034 /** \addtogroup Acq_adapters acquisition_adapters
00035  * Wrap GNU Radio acquisition blocks with an AcquisitionInterface
00036  * \{ */
00037
00038
00039 class ConfigurationInterface;
00040
00041 /*!
00042  * \brief This class adapts a PCPS acquisition block to an AcquisitionInterface
00043  */
00044 class PcpsAcquisitionAdapter : public AcquisitionInterface
00045 {
00046 public:
00047     PcpsAcquisitionAdapter(
00048         const ConfigurationInterface* configuration,
00049         const std::string& role,
00050         const std::string& implementation,
00051         unsigned int in_streams,
00052         unsigned int out_streams,
00053         signal_flag sig_flag);
00054
00055     ~PcpsAcquisitionAdapter() = default;
00056
00057     inline std::string role() override
00058     {
00059         return role_;
00060     }
00061
00062     inline std::string implementation() override
00063     {
00064         return implementation_;
00065     }
00066
00067     inline size_t item_size() override
00068     {
00069         return acq_parameters_.it_size;
00070     }
00071
00072     void connect(gr::top_block_sptr top_block) override;
00073     void disconnect(gr::top_block_sptr top_block) override;
00074     gr::basic_block_sptr get_left_block() override;
00075     gr::basic_block_sptr get_right_block() override;
00076
00077     /*!
00078      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00079      * to efficiently exchange synchronization data between acquisition and
00080      *  tracking blocks
00081      */
00082     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00083
00084     /*!
00085      * \brief Set acquisition channel unique ID
00086      */
00087     inline void set_channel(unsigned int channel) override
00088     {
00089         acquisition_->set_channel(channel);
00090     }
00091
00092     /*!
00093      * \brief Set channel fsm associated to this acquisition instance
00094      */
00095     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00096     {
00097         acquisition_->set_channel_fsm(std::move(channel_fsm));
00098     }
00099
00100     /*!
00101      * \brief Set Doppler center for the grid search
00102      */
00103     void set_doppler_center(int doppler_center) override;
```

```
00104
00105     /*!
00106      * \brief Set the number of Doppler bins to search, centered on
00107      * set_doppler_center() -- see pcps_acquisition::set_doppler_num_bins().
00108      */
00109     void set_doppler_num_bins(unsigned int doppler_num_bins) override;
00110
00111     /*!
00112      * \brief Returns the maximum peak of grid search
00113      */
00114     signed int mag() override;
00115
00116     /*!
00117      * \brief Restart acquisition algorithm
00118      */
00119     void reset() override;
00120
00121     /*!
00122      * \brief Stop running acquisition
00123      */
00124     void stop_acquisition() override;
00125
00126     /*!
00127      * \brief Sets the resampler latency to account it in the acquisition code delay estimation
00128      */
00129     void set_resampler_latency(uint32_t latency_samples) override;
00130
00131     /*!
00132      * \brief Sets local code
00133      */
00134     void set_local_code() override;
00135
00136 private:
00137     const Acq_Conf acq_parameters_;
00138     const signal_flag sig_flag_;
00139     const std::string role_;
00140     const std::string implementation_;
00141     gr::blocks::float_to_complex::sptr float_to_complex_;
00142     complex_byte_to_float_x2_sptr cbyte_to_float_x2_;
00143     Gnss_Synchro* gnss_synchro_;
00144     volk_gnsssdr::vector<std::complex<float» code_;
00145     pcps_acquisition_sptr acquisition_;
00146 };
00147
00148
00149 /** \} */
00150 /** \} */
00151 #endif  // GNSS_SDR_PCPS_ACQUISITION_ADAPTER_H
```

13.3 pcps_acquisition_adapter_custom.h

```
00001 /*!
00002  * \file base_ca_pcps_acquisition_custom.h
00003  * \brief Adapts a PCPS acquisition block to an AcquisitionInterface
00004  * \authors <ul>
00005  *          <li> Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com
00006  *          </ul>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_PCPS_ACQUISITION_ADAPTER_CUSTOM_H
00020 #define GNSS_SDR_PCPS_ACQUISITION_ADAPTER_CUSTOM_H
00021
00022 #include "acquisition_impl_interface.h"
00023 #include "channel_fsm.h"
00024 #include "gnss_synchro.h"
00025 #include "pcps_acquisition.h"
00026 #include "signal_flag.h"
00027 #include <gnuradio/blocks/stream_to_vector.h>
00028 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>
00029
00030 /** \addtogroup Acquisition
00031  * Classes for GNSS signal acquisition
00032  * \{ */
00033 /** \addtogroup Acq_adapters acquisition_adapters
```

```
00034  * Wrap GNU Radio acquisition blocks with an AcquisitionInterface
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class adapts a PCPS acquisition block to an AcquisitionInterface
00042  */
00043 class PcpsAcquisitionAdapterCustom : public AcquisitionInterface
00044 {
00045 public:
00046     PcpsAcquisitionAdapterCustom(const ConfigurationInterface* configuration,
00047         const std::string& role,
00048         const std::string& implementation,
00049         unsigned int in_streams,
00050         unsigned int out_streams,
00051         signal_flag sig_flag);
00052
00053     ~PcpsAcquisitionAdapterCustom() = default;
00054
00055     inline std::string role() override { return role_; }
00056
00057     inline std::string implementation() override { return implementation_; }
00058
00059     inline size_t item_size() override { return item_size_; }
00060
00061     void connect(gr::top_block_sptr top_block) override;
00062     void disconnect(gr::top_block_sptr top_block) override;
00063     gr::basic_block_sptr get_left_block() override;
00064     gr::basic_block_sptr get_right_block() override;
00065
00066     /*!
00067      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00068      * to efficiently exchange synchronization data between acquisition and
00069      *  tracking blocks
00070      */
00071     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00072
00073     /*!
00074      * \brief Set acquisition channel unique ID
00075      */
00076     void set_channel(unsigned int channel) override;
00077
00078     /*!
00079      * \brief Set channel fsm associated to this acquisition instance
00080      */
00081     void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override;
00082
00083     /*!
00084      * \brief Returns the maximum peak of grid search
00085      */
00086     signed int mag() override;
00087
00088     /*!
00089      * \brief Restart acquisition algorithm
00090      */
00091     void reset() override;
00092
00093     /*!
00094      * \brief Stop running acquisition
00095      */
00096     void stop_acquisition() override;
00097
00098     void set_resampler_latency(uint32_t /*latency_samples*/) override {};
00099
00100     /*!
00101      * \brief Sets local code
00102      */
00103     void set_local_code() override;
00104
00105
00106 private:
00107     /*!
00108      * \brief Generate code
00109      */
00110     void code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
    sampling_freq);
00111
00112     const Acq_Conf acq_parameters_;
00113     acquisition_impl_interface_sptr acquisition_cc_;
00114     Gnss_Synchro* gnss_synchro_;
00115     unsigned int channel_;
00116     volk_gnsssdr::vector<std::complex<float>> code_;
00117     gr::blocks::stream_to_vector::sptr stream_to_vector_;
00118     const signal_flag sig_flag_;
00119     const std::string role_;
```

```
00120     const std::string implementation_;
00121     const bool is_type_gr_complex_;
00122     const size_t item_size_;
00123     const bool use_stream_to_vector_;
00124 };
00125
00126
00127 /** \} */
00128 /** \} */
00129 #endif  // GNSS_SDR_PCPS_ACQUISITION_ADAPTER_CUSTOM_H
```

13.4 pcps_acquisition_adapter_fpga.h File Reference

Adapts an FPGA-offloaded PCPS acquisition block to an AcquisitionInterface.

```
#include "acq_conf_fpga.h"
#include "acquisition_interface.h"
#include "channel_fsm.h"
#include "gnss_synchro.h"
#include "pcps_acquisition_fpga.h"
#include "signal_flag.h"
#include <gnuradio/runtime_types.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstddef>
#include <cstdint>
#include <memory>
#include <string>
```

Classes

- class PcpsAcquisitionAdapterFpga

13.4.1 Detailed Description

Adapts an FPGA-offloaded PCPS acquisition block to an AcquisitionInterface.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_acquisition_adapter_fpga.h.

13.5 pcps_acquisition_adapter_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_acquisition_adapter_fpga.h
00003  * \brief Adapts an FPGA-offloaded PCPS acquisition block to an AcquisitionInterface
00004  *
00005  * -----------------------------------------------------------------------------
00006  *
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00011  * SPDX-License-Identifier: GPL-3.0-or-later
00012  *
00013  * -----------------------------------------------------------------------------
00014  */
00015
00016
00017 #ifndef GNSS_SDR_PCPS_ACQUISITION_ADAPTER_FPGA_H
00018 #define GNSS_SDR_PCPS_ACQUISITION_ADAPTER_FPGA_H
00019
00020 #include "acq_conf_fpga.h"
00021 #include "acquisition_interface.h"
00022 #include "channel_fsm.h"
00023 #include "gnss_synchro.h"
00024 #include "pcps_acquisition_fpga.h"
00025 #include "signal_flag.h"
00026 #include <gnuradio/runtime_types.h>
00027 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>
```

```
00028 #include <cstddef>
00029 #include <cstdint>
00030 #include <memory>
00031 #include <string>
00032
00033 class ConfigurationInterface;
00034
00035 class PcpsAcquisitionAdapterFpga : public AcquisitionInterface
00036 {
00037 public:
00038     PcpsAcquisitionAdapterFpga(
00039         const ConfigurationInterface* configuration,
00040         const std::string& role,
00041         const std::string& implementation,
00042         unsigned int in_streams,
00043         unsigned int out_streams,
00044         signal_flag sig_flag);
00045
00046     ~PcpsAcquisitionAdapterFpga() override = default;
00047
00048     inline std::string role() override { return role_; }
00049     inline std::string implementation() override { return implementation_; }
00050     inline size_t item_size() override { return sizeof(int16_t); }
00051
00052     void connect(gr::top_block_sptr top_block) override;
00053     void disconnect(gr::top_block_sptr top_block) override;
00054     gr::basic_block_sptr get_left_block() override;
00055     gr::basic_block_sptr get_right_block() override;
00056
00057     signed int mag() override;
00058     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00059     void set_channel(unsigned int channel) override;
00060     void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override;
00061     void set_doppler_center(int doppler_center) override;
00062     void reset() override;
00063     void stop_acquisition() override;
00064     void set_resampler_latency(uint32_t latency_samples __attribute__((unused))) override {}
00065     void set_local_code() override;
00066
00067 private:
00068     void generate_prn_codes(const ConfigurationInterface* configuration);
00069     void init();
00070
00071     volk_gnsssdr::vector<uint32_t> d_all_fft_codes_;
00072     Acq_Conf_Fpga acq_parameters_;
00073     pcps_acquisition_fpga_sptr acquisition_fpga_;
00074     const signal_flag sig_flag_;
00075     const std::string role_;
00076     const std::string implementation_;
00077 };
00078
00079 #endif  // GNSS_SDR_PCPS_ACQUISITION_ADAPTER_FPGA_H
```

13.6 acquisition_impl_interface.h File Reference

Header file of the interface to an acquisition implementation GNSS block.

```
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <complex>
#include <memory>
```

Classes

- class acquisition_impl_interface

  This abstract class represents an interface to an acquisition GNSS block.

Typedefs

- using acquisition_impl_interface_sptr = gnss_shared_ptr<acquisition_impl_interface>

13.6.1 Detailed Description

Header file of the interface to an acquisition implementation GNSS block.

Author

Mathieu Favraeu, 2025. favreau.mathieu(at)hotmail.com

This header file contains the interface to an abstract class for acquisition algorithms. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file acquisition_impl_interface.h.

13.7 acquisition_impl_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file acquisition_impl_interface.h
00003  * \brief Header file of the interface to an acquisition implementation GNSS block.
00004  * \author Mathieu Favraeu, 2025. favreau.mathieu(at)hotmail.com
00005  *
00006  * This header file contains the interface to an abstract class
00007  * for acquisition algorithms. Since all its methods are virtual,
00008  * this class cannot be instantiated directly, and a subclass can only be
00009  * instantiated directly if all inherited pure virtual methods have been
00010  * implemented by that class or a parent class.
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_ACQUISITION_IMPL_INTERFACE_H
00024 #define GNSS_SDR_ACQUISITION_IMPL_INTERFACE_H
00025
00026 #include "gnss_block_interface.h"
00027 #include "gnss_synchro.h"
00028 #include <gnuradio/block.h>
00029 #include <complex>
00030 #include <memory>
00031
00032 /** \addtogroup Core
00033  * \{ */
00034 /** \addtogroup GNSS_Block_Interfaces GNSS block interfaces
00035  * GNSS block interfaces.
00036  * \{ */
00037
00038 class ChannelFsm;
00039 class acquisition_impl_interface;
00040
00041 using acquisition_impl_interface_sptr = gnss_shared_ptr<acquisition_impl_interface>;
00042
00043 /*! \brief This abstract class represents an interface to an acquisition GNSS block.
00044  *
00045  * Abstract class for acquisition algorithms. Since all its methods are virtual,
00046  * this class cannot be instantiated directly, and a subclass can only be
00047  * instantiated directly if all inherited pure virtual methods have been
00048  * implemented by that class or a parent class.
00049  */
00050 class acquisition_impl_interface : public gr::block
00051 {
00052 public:
00053     acquisition_impl_interface(const std::string& name,
00054         gr::io_signature::sptr input_signature,
00055         gr::io_signature::sptr output_signature) : gr::block(name, std::move(input_signature),
      std::move(output_signature)) {}
00056
00057     virtual void set_gnss_synchro(Gnss_Synchro* gnss_synchro) = 0;
00058     virtual void set_channel(uint32_t channel_id) = 0;
00059     virtual void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) = 0;
00060     virtual void set_local_code(std::complex<float>* /*code*/) {};
00061     virtual void set_local_code(std::complex<float>* /*code_data*/, std::complex<float>*
      /*code_pilot*/) {};
00062     virtual uint32_t mag() const = 0;
00063     virtual void set_active(bool active) = 0;
00064 };
00065
00066
```

```
00067 /** \} */
00068 /** \} */
00069 #endif  // GNSS_SDR_ACQUISITION_INTERFACE */
```

13.8 galileo_e5a_noncoherent_iq_acquisition_caf_cc.h File Reference

Adapts a PCPS acquisition block to an AcquisitionInterface for Galileo E5a data and pilot Signals.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class galileo_e5a_noncoherentIQ_acquisition_caf_cc

  This class implements a Parallel Code Phase Search Acquisition.

Typedefs

- using galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr = gnss_shared_ptr<galileo_e5a_noncoherentIQ_acquisition_caf_c

Functions

- galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr **galileo_e5a_noncoherentIQ_make_acquisition_↩ caf_cc** (const Acq_Conf &conf, bool both_signal_components, int CAF_window_hz, int Zero_padding)

13.8.1 Detailed Description

Adapts a PCPS acquisition block to an AcquisitionInterface for Galileo E5a data and pilot Signals.

Author

Marc Sales, 2014. marcsales92(at)gmail.com \based on work from:

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com
- Marc Molina, 2013. marc.molina.pena@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h.

13.9 galileo_e5a_noncoherent_iq_acquisition_caf_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e5a_noncoherent_iq_acquisition_caf_cc.h
00003  * \brief Adapts a PCPS acquisition block to an AcquisitionInterface for
00004  *  Galileo E5a data and pilot Signals
00005  * \author Marc Sales, 2014. marcsales92(at)gmail.com
00006  * \based on work from:
00007  *          <ul>
```

```
00008  *          <li> Javier Arribas, 2011. jarribas(at)cttc.es
00009  *          <li> Luis Esteve, 2012. luis(at)epsilon-formacion.com
00010  *          <li> Marc Molina, 2013. marc.molina.pena@gmail.com
00011  *          </ul>
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_GALILEO_E5A_NONCOHERENT_IQ_ACQUISITION_CAF_CC_H
00025 #define GNSS_SDR_GALILEO_E5A_NONCOHERENT_IQ_ACQUISITION_CAF_CC_H
00026
00027 #include "acq_conf.h"
00028 #include "acquisition_impl_interface.h"
00029 #include "channel_fsm.h"
00030 #include "gnss_sdr_fft.h"
00031 #include "gnss_synchro.h"
00032 #include <gnuradio/block.h>
00033 #include <gnuradio/gr_complex.h>
00034 #include <fstream>
00035 #include <memory>
00036 #include <string>
00037 #include <utility>
00038 #include <vector>
00039
00040 /** \addtogroup Acquisition
00041  * \{ */
00042 /** \addtogroup Acq_gnuradio_blocks
00043  * \{ */
00044
00045
00046 class galileo_e5a_noncoherentIQ_acquisition_caf_cc;
00047
00048 using galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr =
      gnss_shared_ptr<galileo_e5a_noncoherentIQ_acquisition_caf_cc>;
00049
00050 galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr galileo_e5a_noncoherentIQ_make_acquisition_caf_cc(
00051     const Acq_Conf& conf,
00052     bool both_signal_components,
00053     int CAF_window_hz,
00054     int Zero_padding);
00055
00056 /*!
00057  * \brief This class implements a Parallel Code Phase Search Acquisition.
00058  *
00059  * Check \ref Navitec2012 "An Open Source Galileo E1 Software Receiver",
00060  * Algorithm 1, for a pseudocode description of this implementation.
00061  */
00062 class galileo_e5a_noncoherentIQ_acquisition_caf_cc : public acquisition_impl_interface
00063 {
00064 public:
00065     /*!
00066      * \brief Default destructor.
00067      */
00068     ~galileo_e5a_noncoherentIQ_acquisition_caf_cc();
00069
00070     /*!
00071      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00072      * to exchange synchronization data between acquisition and tracking blocks.
00073      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00074      */
00075     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00076     {
00077         d_gnss_synchro = p_gnss_synchro;
00078     }
00079
00080     /*!
00081      * \brief Returns the maximum peak of grid search.
00082      */
00083     inline unsigned int mag() const override
00084     {
00085         return d_mag;
00086     }
00087
00088     /*!
00089      * \brief Sets local code for PCPS acquisition algorithm.
00090      * \param code - Pointer to the PRN code.
00091      */
00092     void set_local_code(std::complex<float>* code, std::complex<float>* codeQ) override;
00093
```

```
00094     /*!
00095      * \brief Starts acquisition algorithm, turning from standby mode to
00096      * active mode
00097      * \param active - bool that activates/deactivates the block.
00098      */
00099     inline void set_active(bool active) override
00100     {
00101         if (!active)
00102             {
00103                 d_state = 0;
00104             }
00105
00106         d_active = active;
00107     }
00108
00109     /*!
00110      * \brief Set acquisition channel unique ID
00111      * \param channel - receiver channel.
00112      */
00113     inline void set_channel(unsigned int channel) override
00114     {
00115         d_channel = channel;
00116     }
00117
00118     /*!
00119      * \brief Set channel fsm associated to this acquisition instance
00120      */
00121     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00122     {
00123         d_channel_fsm = std::move(channel_fsm);
00124     }
00125
00126     /*!
00127      * \brief Parallel Code Phase Search Acquisition signal processing.
00128      */
00129     int general_work(int noutput_items, gr_vector_int& ninput_items,
00130         gr_vector_const_void_star& input_items,
00131         gr_vector_void_star& output_items) override;
00132
00133 private:
00134     friend galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr
00135     galileo_e5a_noncoherentIQ_make_acquisition_caf_cc(
00136         const Acq_Conf& conf,
00137         bool both_signal_components,
00138         int CAF_window_hz,
00139         int Zero_padding);
00140
00141     galileo_e5a_noncoherentIQ_acquisition_caf_cc(
00142         const Acq_Conf& conf,
00143         bool both_signal_components,
00144         int CAF_window_hz,
00145         int Zero_padding);
00146
00147     void calculate_magnitudes(gr_complex* fft_begin, int doppler_shift,
00148         int doppler_offset);
00149
00150     float estimate_input_power(gr_complex* in);
00151
00152     std::string d_satellite_str;
00153
00154     const Acq_Conf d_acq_params;
00155     std::ofstream d_dump_file;
00156
00157     Gnss_Synchro* d_gnss_synchro;
00158
00159     uint64_t d_sample_counter;
00160
00161     float d_mag;
00162     float d_input_power;
00163     float d_test_statistics;
00164
00165     int d_state;
00166     const int d_CAF_window_hz;
00167     int d_buffer_count;
00168     int d_doppler_resolution;
00169     const int d_fft_size;
00170     int d_num_doppler_bins;
00171     unsigned int d_gr_stream_buffer;
00172     unsigned int d_channel;
00173     unsigned int d_well_count;
00174     unsigned int d_sampled_ms;
00175     unsigned int d_code_phase;
00176
00177     bool d_active;
00178     const bool d_both_signal_components;
00179
00180     std::weak_ptr<ChannelFsm> d_channel_fsm;
```

```
00181     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00182     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00183
00184     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00185     std::vector<gr_complex> d_fft_code_I_A;
00186     std::vector<gr_complex> d_fft_code_I_B;
00187     std::vector<gr_complex> d_fft_code_Q_A;
00188     std::vector<gr_complex> d_fft_code_Q_B;
00189     std::vector<gr_complex> d_inbuffer;
00190     std::vector<float> d_magnitudeIA;
00191     std::vector<float> d_magnitudeIB;
00192     std::vector<float> d_magnitudeQA;
00193     std::vector<float> d_magnitudeQB;
00194     std::vector<float> d_CAF_vector;
00195     std::vector<float> d_CAF_vector_I;
00196     std::vector<float> d_CAF_vector_Q;
00197 };
00198
00199
00200 /** \} */
00201 /** \} */
00202 #endif  // GNSS_SDR_GALILEO_E5A_NONCOHERENT_IQ_ACQUISITION_CAF_CC_H
```

13.10 galileo_pcps_8ms_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class galileo_pcps_8ms_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).

Typedefs

- using galileo_pcps_8ms_acquisition_cc_sptr = gnss_shared_ptr<galileo_pcps_8ms_acquisition_cc>

Functions

- galileo_pcps_8ms_acquisition_cc_sptr **galileo_pcps_8ms_make_acquisition_cc** (const Acq_Conf &conf)

13.10.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition for Galileo E1 signals with coherent integration time = 8 ms (two codes).

Author

Marc Molina, 2013. marc.molina.pena(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_pcps_8ms_acquisition_cc.h.

13.11 galileo_pcps_8ms_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_pcps_8ms_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition for
00004  * Galileo E1 signals with coherent integration time = 8 ms (two codes)
00005  * \author Marc Molina, 2013. marc.molina.pena(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_PCPS_8MS_ACQUISITION_CC_H
00019 #define GNSS_SDR_PCPS_8MS_ACQUISITION_CC_H
00020
00021 #include "acq_conf.h"
00022 #include "acquisition_impl_interface.h"
00023 #include "channel_fsm.h"
00024 #include "gnss_sdr_fft.h"
00025 #include "gnss_synchro.h"
00026 #include <gnuradio/block.h>
00027 #include <gnuradio/gr_complex.h>
00028 #include <fstream>
00029 #include <memory>
00030 #include <string>
00031 #include <utility>
00032 #include <vector>
00033
00034 /** \addtogroup Acquisition
00035  * \{ */
00036 /** \addtogroup Acq_gnuradio_blocks
00037  * \{ */
00038
00039
00040 class galileo_pcps_8ms_acquisition_cc;
00041
00042 using galileo_pcps_8ms_acquisition_cc_sptr = gnss_shared_ptr<galileo_pcps_8ms_acquisition_cc>;
00043
00044 galileo_pcps_8ms_acquisition_cc_sptr
00045 galileo_pcps_8ms_make_acquisition_cc(const Acq_Conf& conf);
00046
00047 /*!
00048  * \brief This class implements a Parallel Code Phase Search Acquisition for
00049  * Galileo E1 signals with coherent integration time = 8 ms (two codes)
00050  */
00051 class galileo_pcps_8ms_acquisition_cc : public acquisition_impl_interface
00052 {
00053 public:
00054     /*!
00055      * \brief Default destructor.
00056      */
00057     ~galileo_pcps_8ms_acquisition_cc();
00058
00059     /*!
00060      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00061      * to exchange synchronization data between acquisition and tracking blocks.
00062      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00063      */
00064     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00065     {
00066         d_gnss_synchro = p_gnss_synchro;
00067     }
00068
00069     /*!
00070      * \brief Returns the maximum peak of grid search.
00071      */
00072     inline uint32_t mag() const override
00073     {
00074         return d_mag;
00075     }
00076
00077     /*!
00078      * \brief Sets local code for PCPS acquisition algorithm.
00079      * \param code - Pointer to the PRN code.
00080      */
00081     void set_local_code(std::complex<float>* code) override;
00082
00083     /*!
```

```
00084      * \brief Starts acquisition algorithm, turning from standby mode to
00085      * active mode
00086      * \param active - bool that activates/deactivates the block.
00087      */
00088     inline void set_active(bool active) override
00089     {
00090         if (!active)
00091             {
00092                 d_state = 0;
00093             }
00094
00095         d_active = active;
00096     }
00097
00098     /*!
00099      * \brief Set acquisition channel unique ID
00100      * \param channel - receiver channel.
00101      */
00102     inline void set_channel(uint32_t channel) override
00103     {
00104         d_channel = channel;
00105     }
00106
00107     /*!
00108      * \brief Set channel fsm associated to this acquisition instance
00109      */
00110     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00111     {
00112         d_channel_fsm = std::move(channel_fsm);
00113     }
00114
00115     /*!
00116      * \brief Parallel Code Phase Search Acquisition signal processing.
00117      */
00118     int general_work(int noutput_items, gr_vector_int& ninput_items,
00119         gr_vector_const_void_star& input_items,
00120         gr_vector_void_star& output_items) override;
00121
00122 private:
00123     friend galileo_pcps_8ms_acquisition_cc_sptr
00124     galileo_pcps_8ms_make_acquisition_cc(const Acq_Conf& conf);
00125
00126     explicit galileo_pcps_8ms_acquisition_cc(const Acq_Conf& conf);
00127
00128     void calculate_magnitudes(
00129         gr_complex* fft_begin,
00130         int32_t doppler_shift,
00131         int32_t doppler_offset);
00132
00133     std::string d_satellite_str;
00134     const Acq_Conf d_acq_params;
00135     std::ofstream d_dump_file;
00136
00137     Gnss_Synchro* d_gnss_synchro;
00138
00139     uint64_t d_sample_counter;
00140
00141     float d_mag;
00142     float d_input_power;
00143     float d_test_statistics;
00144     int32_t d_state;
00145     uint32_t d_channel;
00146     uint32_t d_well_count;
00147     const uint32_t d_fft_size;
00148     uint32_t d_num_doppler_bins;
00149     uint32_t d_code_phase;
00150
00151     bool d_active;
00152
00153     std::weak_ptr<ChannelFsm> d_channel_fsm;
00154     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00155     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00156
00157     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00158     std::vector<gr_complex> d_fft_code_A;
00159     std::vector<gr_complex> d_fft_code_B;
00160     std::vector<float> d_magnitude;
00161 };
00162
00163
00164 /** \} */
00165 /** \} */
00166 #endif  // GNSS_SDR_PCPS_8MS_ACQUISITION_CC_H
```

13.12 pcps_acquisition.h File Reference

This class implements a Parallel Code Phase Search Acquisition.

```
#include "acquisition_impl_interface.h"
#include "acq_conf.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include <armadillo>
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <gnuradio/thread/thread.h>
#include <gnuradio/types.h>
#include <volk/volk_complex.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <complex>
#include <cstdint>
#include <memory>
#include <queue>
#include <string>
#include <utility>
#include <gsl-lite/gsl-lite.hpp>
```

Classes

- class pcps_acquisition

  This class implements a Parallel Code Phase Search Acquisition.

Typedefs

- using pcps_acquisition_sptr = gnss_shared_ptr<pcps_acquisition>

Functions

- pcps_acquisition_sptr **pcps_make_acquisition** (const Acq_Conf &conf_)

13.12.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition.
Acquisition strategy (Kay Borre book + CFAR threshold).

1. Compute the input signal power estimation
2. Doppler serial search loop
3. Perform the FFT-based circular convolution (parallel time search)
4. Record the maximum peak and the associated synchronization parameters
5. Compute the test statistics and compare to the threshold
6. Declare positive or negative acquisition using a message queue

Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkhauser, 2007. pp 81-84

Authors

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com
- Marc Molina, 2013. marc.molina.pena@gmail.com
- Cillian O'Driscoll, 2017. cillian(at)ieee.org
- Antonio Ramos, 2017. antonio.ramos@cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_acquisition.h.

13.13 pcps_acquisition.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_acquisition.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition
00004  *
00005  *  Acquisition strategy (Kay Borre book + CFAR threshold).
00006  *  <ol>
00007  *  <li> Compute the input signal power estimation
00008  *  <li> Doppler serial search loop
00009  *  <li> Perform the FFT-based circular convolution (parallel time search)
00010  *  <li> Record the maximum peak and the associated synchronization parameters
00011  *  <li> Compute the test statistics and compare to the threshold
00012  *  <li> Declare positive or negative acquisition using a message queue
00013  *  </ol>
00014  *
00015  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00016  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00017  * Approach", Birkhauser, 2007. pp 81-84
00018  *
00019  * \authors <ul>
00020  *          <li> Javier Arribas, 2011. jarribas(at)cttc.es
00021  *          <li> Luis Esteve, 2012. luis(at)epsilon-formacion.com
00022  *          <li> Marc Molina, 2013. marc.molina.pena@gmail.com
00023  *          <li> Cillian O'Driscoll, 2017. cillian(at)ieee.org
00024  *          <li> Antonio Ramos, 2017. antonio.ramos@cttc.es
00025  *          </ul>
00026  *
00027  * -----------------------------------------------------------------------------
00028  *
00029  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00030  * This file is part of GNSS-SDR.
00031  *
00032  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00033  * SPDX-License-Identifier: GPL-3.0-or-later
00034  *
00035  * -----------------------------------------------------------------------------
00036  */
00037
00038 #ifndef GNSS_SDR_PCPS_ACQUISITION_H
00039 #define GNSS_SDR_PCPS_ACQUISITION_H
00040
00041 #include "acquisition_impl_interface.h"
00042 #if ARMA_NO_BOUND_CHECKING
00043 #define ARMA_NO_DEBUG 1
00044 #endif
00045
00046 #include "acq_conf.h"
00047 #include "channel_fsm.h"
00048 #include "gnss_sdr_fft.h"
00049 #if CUDA_GPU_ACCEL
00050 #include "cuda_pcps_engine.h"
00051 #endif
00052 #include <armadillo>
00053 #include <gnuradio/block.h>
00054 #include <gnuradio/gr_complex.h>              // for gr_complex
00055 #include <gnuradio/thread/thread.h>           // for scoped_lock
00056 #include <gnuradio/types.h>                   // for gr_vector_const_void_star
00057 #include <volk/volk_complex.h>                // for lv_16sc_t
00058 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00059 #include <complex>
00060 #include <cstdint>
00061 #include <memory>
00062 #include <queue>
00063 #include <string>
00064 #include <utility>
```

```
00065
00066
00067 #if HAS_STD_SPAN
00068 #include <span>
00069 namespace own = std;
00070 #else
00071 #include <gsl-lite/gsl-lite.hpp>
00072 namespace own = gsl_lite;
00073 #endif
00074
00075 /** \addtogroup Acquisition
00076  * Classes for GNSS signal acquisition
00077  * \{ */
00078 /** \addtogroup Acq_gnuradio_blocks acquisition_gr_blocks
00079  * GNU Radio processing blocks for GNSS signal acquisition
00080  * \{ */
00081
00082
00083 class Gnss_Synchro;
00084 class pcps_acquisition;
00085
00086 using pcps_acquisition_sptr = gnss_shared_ptr<pcps_acquisition>;
00087
00088 pcps_acquisition_sptr pcps_make_acquisition(const Acq_Conf& conf_);
00089
00090 /*!
00091  * \brief This class implements a Parallel Code Phase Search Acquisition.
00092  *
00093  * Check \ref Navitec2012 "An Open Source Galileo E1 Software Receiver",
00094  * Algorithm 1, for a pseudocode description of this implementation.
00095  */
00096 class pcps_acquisition : public acquisition_impl_interface
00097 {
00098 public:
00099     ~pcps_acquisition() noexcept override;
00100
00101     /*!
00102      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00103      * to exchange synchronization data between acquisition and tracking blocks.
00104      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00105      */
00106     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00107     {
00108         gr::thread::scoped_lock lock(d_setlock);  // require mutex with work function called by the
      scheduler
00109         d_gnss_synchro = p_gnss_synchro;
00110     }
00111
00112     /*!
00113      * \brief Sets local code for PCPS acquisition algorithm.
00114      * \param code - Pointer to the PRN code.
00115      */
00116     void set_local_code(std::complex<float>* code) override;
00117
00118     void set_resampler_latency(uint32_t latency_samples);
00119
00120     /*!
00121      * \brief Returns the maximum peak of grid search.
00122      */
00123     inline uint32_t mag() const override
00124     {
00125         return 0;  // Not implemented
00126     }
00127
00128     /*!
00129      * \brief Starts acquisition algorithm, turning from standby mode to
00130      * active mode
00131      * \param active - bool that activates/deactivates the block.
00132      */
00133     void set_active(bool active) override;
00134
00135     /*!
00136      * \brief Set acquisition channel unique ID
00137      * \param channel - receiver channel.
00138      */
00139     inline void set_channel(uint32_t channel) override
00140     {
00141         d_channel = channel;
00142     }
00143
00144     /*!
00145      * \brief Set channel fsm associated to this acquisition instance
00146      */
00147     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00148     {
00149         d_channel_fsm = std::move(channel_fsm);
00150     }
```

```
00151
00152     /*!
00153      * \brief Set Doppler center frequency for the grid search. It will refresh the Doppler grid.
00154      * \param doppler_center - Frequency center of the search grid [Hz].
00155      */
00156     void set_doppler_center(int32_t doppler_center);
00157
00158     /*!
00159      * \brief Sets how many Doppler bins to search, centered on set_doppler_center().
00160      * Refreshes the Doppler grid and updates the detection threshold when the
00161      * bin count changes, supporting both unassisted and assisted searches.
00162      * \param num_doppler_bins - candidate bin count, capped at the full grid size.
00163      * Use 0 for the full range defined by doppler_max and doppler_step, 1 for
00164      * a known Doppler, or larger counts for partial uncertainty.
00165      */
00166     void set_doppler_num_bins(uint32_t num_doppler_bins);
00167
00168     //! Whether the CUDA engine is available. Inspect only while acquisition is stopped.
00169     inline bool cuda_ready() const
00170     {
00171 #if CUDA_GPU_ACCEL
00172         return d_cuda_engine != nullptr;
00173 #else
00174         return false;
00175 #endif
00176     }
00177
00178     //! Completed CUDA grids over the block's lifetime, excluding warm-up.
00179     //! Inspect only while acquisition is stopped.
00180     inline uint64_t cuda_grid_count() const
00181     {
00182 #if CUDA_GPU_ACCEL
00183         return d_cuda_grid_count;
00184 #else
00185         return 0;
00186 #endif
00187     }
00188
00189     /*!
00190      * \brief Parallel Code Phase Search Acquisition signal processing.
00191      */
00192     int general_work(int noutput_items, gr_vector_int& ninput_items,
00193         gr_vector_const_void_star& input_items,
00194         gr_vector_void_star& output_items) override;
00195
00196 private:
00197     friend pcps_acquisition_sptr pcps_make_acquisition(const Acq_Conf& conf_);
00198     explicit pcps_acquisition(const Acq_Conf& conf_);
00199
00200     struct AcquisitionResult
00201     {
00202         int32_t doppler{0};
00203         uint32_t index_time{0};
00204         uint64_t sample_count{0};
00205         float test_statistics{0};
00206         bool positive_acq{false};
00207     };
00208
00209     // Narrowed grids always need reference rows; full grids need them only
00210     // for CFAR when wraparound separation is insufficient.
00211     // Call after construction: d_use_CFAR_algorithm_flag is initialized later.
00212     bool needs_extra_reference_row(uint32_t candidate_bins) const;
00213
00214     void update_local_carrier(own::span<gr_complex> carrier_vector, float freq) const;
00215     void update_grid_doppler_wipeoffs();
00216     void update_grid_doppler_wipeoffs_step2();
00217     void doppler_grid(const gr_complex* in);
00218     void doppler_grid_cpu(const gr_complex* in);
00219 #if CUDA_GPU_ACCEL
00220     void init_cuda_engine();
00221     void cuda_upload_wipeoffs(CudaPcpsEngine::GridId grid);
00222     bool doppler_grid_cuda(const gr_complex* in);
00223 #endif
00224     AcquisitionResult compute_statistics();
00225     void update_synchro(const AcquisitionResult& result);
00226     void handle_threshold_reached(AcquisitionResult& result);
00227     void handle_integration_done(const AcquisitionResult& result);
00228     void acquisition_core(uint64_t sample_count);
00229     void log_acquisition(const AcquisitionResult& result) const;
00230     void send_negative_acquisition(const AcquisitionResult& result);
00231     void send_positive_acquisition(const AcquisitionResult& result);
00232     void dump_results(const AcquisitionResult& result);
00233     void ensure_dump_grid_allocated();
00234     void copy_magnitude_grid_to_dump_grid();
00235     bool should_dump_channel() const;
00236     std::complex<float>* doppler_wipeoff_data(uint32_t doppler_index);
00237     std::complex<float>* doppler_wipeoff_step_two_data(uint32_t doppler_index);
```

```
00238     float* magnitude_grid_data(uint32_t doppler_index);
00239     const float* magnitude_grid_data(uint32_t doppler_index) const;
00240     bool is_fdma();
00241     float get_threshold() const;
00242     // candidate_count excludes trailing noise-reference rows.
00243     // CFAR also takes the total computed row count for reference lookup.
00244     AcquisitionResult first_vs_second_peak_statistic(uint32_t candidate_count, int32_t doppler_max,
      int32_t doppler_step);
00245     AcquisitionResult max_to_input_power_statistic(uint32_t num_doppler_bins, uint32_t
      candidate_count, int32_t doppler_max, int32_t doppler_step);
00246     void wait_if_active();
00247
00248     const Acq_Conf d_acq_parameters;
00249     const std::string d_dump_filename;
00250     const float d_doppler_max;
00251     const uint32_t d_samplesPerChip;
00252     const uint32_t d_doppler_step;
00253     const uint32_t d_samples_to_consume;
00254     // Exact (true) dwell length minus d_samples_to_consume, in [0, 1) samples.
00255     const double d_dwell_residual_samples;
00256     const uint32_t d_fft_size;
00257     const uint32_t d_effective_fft_size;
00258     const uint32_t d_magnitude_grid_stride;
00259     const uint32_t d_doppler_wipeoffs_stride;
00260     const uint32_t d_num_doppler_bins;
00261     const uint32_t d_num_doppler_bins_step2;
00262     const uint32_t d_dump_channel;
00263     // CFAR reference separation target (Hz): (sidelobes + 0.5) / integration time.
00264     // Selects extra reference rows; never extends them beyond d_doppler_max.
00265     const float d_min_reference_separation_hz;
00266     // Full-grid CFAR wraparound separation is insufficient.
00267     // Computed from conf_ during construction, before the runtime flag exists.
00268     const bool d_full_grid_reference_needs_extra_row;
00269     // Full-grid candidate count plus reference rows; restored by set_doppler_num_bins(0).
00270     const uint32_t d_num_doppler_bins_full_grid_active;
00271     // Capacity includes two reference rows, even when only narrowed grids need them.
00272     const uint32_t d_num_doppler_bins_capacity;
00273     const float d_threshold_step_two;
00274     const bool d_cshort;
00275     const bool d_use_CFAR_algorithm_flag;
00276     const bool d_dump;
00277
00278     // Need lock to access these
00279     std::weak_ptr<ChannelFsm> d_channel_fsm;
00280     std::unique_ptr<gr::thread::thread> d_worker;
00281     Gnss_Synchro* d_gnss_synchro;
00282     std::queue<Gnss_Synchro> d_monitor_queue;
00283     int32_t d_state;
00284     int32_t d_doppler_center;
00285     int32_t d_doppler_bias;
00286     // Active candidate and reference rows; bounded by d_num_doppler_bins_capacity.
00287     uint32_t d_num_doppler_bins_active;
00288     // Trailing noise-reference rows: 0 if unnecessary, 1 at center + doppler_max
00289     // for a single candidate, otherwise 2 at center +/- doppler_max.
00290     // CFAR uses the reference opposite the winning candidate.
00291     uint32_t d_num_reference_rows_active;
00292     // CFAR threshold for active candidates, excluding reference rows.
00293     // Recompute when the candidate count changes to preserve the requested PFA.
00294     float d_threshold_active;
00295     uint32_t d_buffer_sample_count;
00296     // DDA/Bresenham accumulator keeping stream consumption aligned with the
00297     // true dwell boundary; see general_work() case 0/1.
00298     double d_dwell_residual_accum;
00299     uint32_t d_pending_skip_samples;
00300     uint32_t d_channel;
00301     uint32_t d_resampler_latency_samples;
00302     uint64_t d_sample_count;
00303     bool d_step_two;
00304     bool d_active;
00305     bool d_worker_active;
00306
00307     // Only access these in acquisition_core and functions strictly called from acquisition_core
00308     uint32_t d_num_noncoherent_integrations_counter;
00309     int64_t d_dump_number;
00310     float d_input_power;
00311     float d_doppler_center_step_two;
00312     volk_gnsssdr::vector<float> d_magnitude_grid;
00313     volk_gnsssdr::vector<float> d_tmp_buffer;
00314     volk_gnsssdr::vector<std::complex<float>> d_input_signal;
00315     volk_gnsssdr::vector<std::complex<float>> d_grid_doppler_wipeoffs_step_two;
00316     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00317     arma::fmat d_grid;
00318     arma::fmat d_narrow_grid;
00319
00320     // These are never accessed outside acquisition_core while acquisition is active
00321     volk_gnsssdr::vector<std::complex<float>> d_grid_doppler_wipeoffs;
00322     volk_gnsssdr::vector<std::complex<float>> d_fft_codes;
```

```
00323     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00324 #if CUDA_GPU_ACCEL
00325     std::unique_ptr<CudaPcpsEngine> d_cuda_engine;  // null => CPU path
00326     uint64_t d_cuda_grid_count{0};
00327 #endif
00328 };
00329
00330
00331 /** \} */
00332 /** \} */
00333 #endif  // GNSS_SDR_PCPS_ACQUISITION_H
```

13.14 pcps_acquisition_fine_doppler_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition with multi-dwells and fine Doppler estimation for GPS L1 C/A signal.

```
#include "acquisition_impl_interface.h"
#include "acq_conf.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <armadillo>
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstdint>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
```

Classes

- class pcps_acquisition_fine_doppler_cc

  This class implements a Parallel Code Phase Search Acquisition.

Typedefs

- using pcps_acquisition_fine_doppler_cc_sptr = gnss_shared_ptr<pcps_acquisition_fine_doppler_cc>

Functions

- pcps_acquisition_fine_doppler_cc_sptr **pcps_make_acquisition_fine_doppler_cc** (const Acq_Conf &conf_)

13.14.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with multi-dwells and fine Doppler estimation for GPS L1 C/A signal.
Acquisition strategy (Kay Borre book).

1. Compute the input signal power estimation
2. Doppler serial search loop
3. Perform the FFT-based circular convolution (parallel time search)
4. Record the maximum peak and the associated synchronization parameters
5. Compute the test statistics and compare to the threshold
6. Declare positive or negative acquisition using a message port

Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkhauser, 2007. pp 81-84

Authors

- Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pcps_acquisition_fine_doppler_cc.h.

13.15 pcps_acquisition_fine_doppler_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_acquisition_fine_doppler_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition with multi-dwells and fine
      Doppler estimation
00004  * for GPS L1 C/A signal
00005  *
00006  *  Acquisition strategy (Kay Borre book).
00007  *  <ol>
00008  *  <li> Compute the input signal power estimation
00009  *  <li> Doppler serial search loop
00010  *  <li> Perform the FFT-based circular convolution (parallel time search)
00011  *  <li> Record the maximum peak and the associated synchronization parameters
00012  *  <li> Compute the test statistics and compare to the threshold
00013  *  <li> Declare positive or negative acquisition using a message port
00014  *  </ol>
00015  *
00016  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00017  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00018  * Approach", Birkhauser, 2007. pp 81-84
00019  *
00020  * \authors <ul>
00021  *          <li> Javier Arribas, 2013. jarribas(at)cttc.es
00022  *          </ul>
00023  *
00024  * -----------------------------------------------------------------------------
00025  *
00026  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00027  * This file is part of GNSS-SDR.
00028  *
00029  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00030  * SPDX-License-Identifier: GPL-3.0-or-later
00031  *
00032  * -----------------------------------------------------------------------------
00033  */
00034
00035 #ifndef GNSS_SDR_PCPS_ACQUISITION_FINE_DOPPLER_CC_H
00036 #define GNSS_SDR_PCPS_ACQUISITION_FINE_DOPPLER_CC_H
00037
00038 #include "acquisition_impl_interface.h"
00039 #if ARMA_NO_BOUND_CHECKING
00040 #define ARMA_NO_DEBUG 1
00041 #endif
00042
00043 #include "acq_conf.h"
00044 #include "channel_fsm.h"
00045 #include "gnss_sdr_fft.h"
00046 #include "gnss_synchro.h"
00047 #include <armadillo>
00048 #include <gnuradio/block.h>
00049 #include <gnuradio/gr_complex.h>
00050 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00051 #include <cstdint>
00052 #include <fstream>
00053 #include <memory>
00054 #include <string>
00055 #include <utility>
00056
00057
00058 /** \addtogroup Acquisition
00059  * \{ */
00060 /** \addtogroup Acq_gnuradio_blocks
00061  * \{ */
00062
00063
00064 class pcps_acquisition_fine_doppler_cc;
00065
00066 using pcps_acquisition_fine_doppler_cc_sptr = gnss_shared_ptr<pcps_acquisition_fine_doppler_cc>;
00067
```

```
00068 pcps_acquisition_fine_doppler_cc_sptr pcps_make_acquisition_fine_doppler_cc(const Acq_Conf& conf_);
00069
00070 /*!
00071  * \brief This class implements a Parallel Code Phase Search Acquisition.
00072  *
00073  */
00074 class pcps_acquisition_fine_doppler_cc : public acquisition_impl_interface
00075 {
00076 public:
00077     /*!
00078      * \brief Default destructor.
00079      */
00080     ~pcps_acquisition_fine_doppler_cc() = default;
00081
00082     /*!
00083      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00084      * to exchange synchronization data between acquisition and tracking blocks.
00085      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00086      */
00087     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00088     {
00089         d_gnss_synchro = p_gnss_synchro;
00090     }
00091
00092     /*!
00093      * \brief Returns the maximum peak of grid search.
00094      */
00095     inline unsigned int mag() const override
00096     {
00097         return d_test_statistics;
00098     }
00099
00100     /*!
00101      * \brief Sets local code for PCPS acquisition algorithm.
00102      * \param code - Pointer to the PRN code.
00103      */
00104     void set_local_code(std::complex<float>* code) override;
00105
00106     /*!
00107      * \brief Starts acquisition algorithm, turning from standby mode to
00108      * active mode
00109      * \param active - bool that activates/deactivates the block.
00110      */
00111     inline void set_active(bool active) override
00112     {
00113         if (!active)
00114             {
00115                 d_state = 0;
00116             }
00117
00118         d_active = active;
00119     }
00120
00121     /*!
00122      * \brief Set acquisition channel unique ID
00123      * \param channel - receiver channel.
00124      */
00125     inline void set_channel(unsigned int channel) override
00126     {
00127         d_channel = channel;
00128         d_dump_channel = d_channel;
00129     }
00130
00131     /*!
00132      * \brief Set channel fsm associated to this acquisition instance
00133      */
00134     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00135     {
00136         d_channel_fsm = std::move(channel_fsm);
00137     }
00138
00139     /*!
00140      * \brief Parallel Code Phase Search Acquisition signal processing.
00141      */
00142     int general_work(int noutput_items, gr_vector_int& ninput_items,
00143         gr_vector_const_void_star& input_items,
00144         gr_vector_void_star& output_items) override;
00145
00146 private:
00147     /*!
00148      * \brief Obtains the next power of 2 greater or equal to the input parameter
00149      * \param n - Integer value to obtain the next power of 2.
00150      */
00151     unsigned int nextPowerOf2(unsigned int n);
00152
00153     void dump_results(int effective_fft_size);
00154
```

```
00155     void forecast(int noutput_items, gr_vector_int& ninput_items_required) override;
00156
00157     friend pcps_acquisition_fine_doppler_cc_sptr pcps_make_acquisition_fine_doppler_cc(const Acq_Conf&
      conf_);
00158     explicit pcps_acquisition_fine_doppler_cc(const Acq_Conf& conf_);
00159
00160     int compute_and_accumulate_grid(gr_vector_const_void_star& input_items);
00161     int estimate_Doppler();
00162     float estimate_input_power(gr_vector_const_void_star& input_items);
00163     float compute_CAF();
00164     void reset_grid();
00165     void update_carrier_wipeoff();
00166     void update_fine_doppler_freq_bins();
00167     bool start() override;
00168
00169     arma::fmat grid_;
00170
00171     std::string d_satellite_str;
00172
00173     const Acq_Conf d_acq_params;
00174     std::string d_dump_filename;
00175
00176     Gnss_Synchro* d_gnss_synchro;
00177
00178     int64_t d_dump_number;
00179     uint64_t d_sample_counter;
00180
00181     float d_test_statistics;
00182
00183     int d_positive_acq;
00184     int d_state;
00185     const int d_num_doppler_points;
00186     int d_well_count;
00187     int d_n_samples_in_buffer;
00188     const int d_fft_size;
00189     const int d_fine_doppler_zero_padding_factor;
00190     const int d_fine_doppler_prn_replicas;
00191     const int d_fine_doppler_signal_samples;
00192     const int d_fine_doppler_fft_size;
00193     int d_gnuradio_forecast_samples;
00194     unsigned int d_channel;
00195     unsigned int d_dump_channel;
00196
00197     bool d_active;
00198     bool d_dump;
00199
00200     std::weak_ptr<ChannelFsm> d_channel_fsm;
00201     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00202     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00203     std::unique_ptr<gnss_fft_complex_fwd> d_fine_doppler_fft;
00204
00205     volk_gnsssdr::vector<volk_gnsssdr::vector<std::complex<float>>> d_grid_doppler_wipeoffs;
00206     volk_gnsssdr::vector<volk_gnsssdr::vector<float>> d_grid_data;
00207     volk_gnsssdr::vector<gr_complex> d_fft_codes;
00208     volk_gnsssdr::vector<gr_complex> d_10_ms_buffer;
00209     volk_gnsssdr::vector<float> d_magnitude;
00210     volk_gnsssdr::vector<gr_complex> d_fine_doppler_code_replica;
00211     volk_gnsssdr::vector<float> d_fine_doppler_magnitude;
00212     volk_gnsssdr::vector<float> d_fine_doppler_freq_bins;
00213 };
00214
00215
00216 /** \} */
00217 /** \} */
00218 #endif  // GNSS_SDR_PCPS_ACQUISITION_FINE_DOPPLER_CC_H
```

13.16 pcps_acquisition_fpga.h File Reference

This class implements a Parallel Code Phase Search Acquisition for the FPGA.

```
#include "acq_conf_fpga.h"
#include "channel_fsm.h"
#include "fpga_acquisition.h"
#include <cstdint>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class pcps_acquisition_fpga

  This class implements a Parallel Code Phase Search Acquisition that uses the FPGA.

Typedefs

- using pcps_acquisition_fpga_sptr = std::shared_ptr<pcps_acquisition_fpga>

Functions

- pcps_acquisition_fpga_sptr **pcps_make_acquisition_fpga** (Acq_Conf_Fpga *conf_, uint32_t acq_buff_↩ num, std::vector< std::pair< uint32_t, uint32_t > > &downsampling_filter_specs, uint32_t &max_FFT_size)

13.16.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition for the FPGA.
Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkhauser, 2007. pp 81-84

Authors

- Marc Majoral, 2019. mmajoral(at)cttc.es
- Javier Arribas, 2019. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_acquisition_fpga.h.

13.17 pcps_acquisition_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_acquisition_fpga.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition for the FPGA
00004  *
00005  *
00006  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00007  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00008  * Approach", Birkhauser, 2007. pp 81-84
00009  *
00010  * \authors <ul>
00011  *          <li> Marc Majoral, 2019. mmajoral(at)cttc.es
00012  *          <li> Javier Arribas, 2019. jarribas(at)cttc.es
00013  *          </ul>
00014  *
00015  * -----------------------------------------------------------------------------
00016  *
00017  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00018  * This file is part of GNSS-SDR.
00019  *
00020  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00021  * SPDX-License-Identifier: GPL-3.0-or-later
00022  *
00023  * -----------------------------------------------------------------------------
00024  */
00025
00026 #ifndef GNSS_SDR_PCPS_ACQUISITION_FPGA_H
00027 #define GNSS_SDR_PCPS_ACQUISITION_FPGA_H
00028
00029 #include "acq_conf_fpga.h"
00030 #include "channel_fsm.h"
00031 #include "fpga_acquisition.h"
00032 #include <cstdint>  // for uint32_t
00033 #include <memory>   // for shared_ptr
00034 #include <string>   // for string
00035 #include <utility>  // for for std::move, std::pair
00036 #include <vector>   // for std::vector
00037
00038 /** \addtogroup Acquisition
00039  * \{ */
00040 /** \addtogroup Acq_gnuradio_blocks
00041  * \{ */
```

```
00042
00043
00044 class Gnss_Synchro;
00045
00046 class pcps_acquisition_fpga;
00047
00048 using pcps_acquisition_fpga_sptr = std::shared_ptr<pcps_acquisition_fpga>;
00049
00050 pcps_acquisition_fpga_sptr pcps_make_acquisition_fpga(Acq_Conf_Fpga* conf_, uint32_t acq_buff_num,
      std::vector<std::pair<uint32_t, uint32_t»& downsampling_filter_specs, uint32_t& max_FFT_size);
00051
00052 /*!
00053  * \brief This class implements a Parallel Code Phase Search Acquisition that uses the FPGA.
00054  *
00055  * Check \ref Navitec2012 "An Open Source Galileo E1 Software Receiver",
00056  * Algorithm 1, for a pseudocode description of this implementation.
00057  */
00058 class pcps_acquisition_fpga
00059 {
00060 public:
00061     /*!
00062      * \brief Destructor
00063      */
00064     ~pcps_acquisition_fpga() = default;
00065
00066     /*!
00067      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00068      * to exchange synchronization data between acquisition and tracking blocks.
00069      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00070      */
00071     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro)
00072     {
00073         d_gnss_synchro = p_gnss_synchro;
00074     }
00075
00076     /*!
00077      * \brief Returns the maximum peak of grid search.
00078      */
00079     inline uint32_t mag() const
00080     {
00081         return d_mag;
00082     }
00083
00084     /*!
00085      * \brief Initializes acquisition algorithm.
00086      */
00087     void init();
00088
00089     /*!
00090      * \brief Sets local code for PCPS acquisition algorithm.
00091      */
00092     void set_local_code();
00093
00094     /*!
00095      * \brief Starts acquisition algorithm, turning from standby mode to
00096      * active mode
00097      * \param active - bool that activates/deactivates the block.
00098      */
00099     void set_active(bool active);
00100
00101     /*!
00102      * \brief Set acquisition channel unique ID
00103      * \param channel - receiver channel.
00104      */
00105     inline void set_channel(uint32_t channel)
00106     {
00107         d_channel = channel;
00108     }
00109
00110     /*!
00111      * \brief Set channel fsm associated to this acquisition instance
00112      */
00113     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm)
00114     {
00115         d_channel_fsm = std::move(channel_fsm);
00116     }
00117
00118     /*!
00119      * \brief Set Doppler center frequency for the grid search. It will refresh the Doppler grid.
00120      * \param doppler_center - Frequency center of the search grid [Hz].
00121      */
00122     void set_doppler_center(int32_t doppler_center);
00123
00124     /*!
00125      * \brief This function triggers a HW reset of the FPGA PL.
00126      */
00127     void reset_acquisition();
```

```
00128
00129     /*!
00130      * \brief stop the acquisition and the other FPGA modules.
00131      */
00132     void stop_acquisition();
00133
00134 private:
00135     friend pcps_acquisition_fpga_sptr pcps_make_acquisition_fpga(Acq_Conf_Fpga* conf, uint32_t
      acq_buff_num, std::vector<std::pair<uint32_t, uint32_t>& downsampling_filter_specs, uint32_t&
      max_FFT_size);
00136     explicit pcps_acquisition_fpga(Acq_Conf_Fpga* conf, uint32_t acq_buff_num,
      std::vector<std::pair<uint32_t, uint32_t>& downsampling_filter_specs, uint32_t& max_FFT_size);
00137
00138     void send_negative_acquisition();
00139     void send_positive_acquisition();
00140     void acquisition_core(uint32_t num_doppler_bins, uint32_t doppler_step, int32_t doppler_min);
00141     float first_vs_second_peak_statistic(uint32_t& indext, int32_t& doppler, uint32_t
      num_doppler_bins, int32_t doppler_max, int32_t doppler_step);
00142
00143     std::shared_ptr<Fpga_Acquisition> d_acquisition_fpga;
00144     std::weak_ptr<ChannelFsm> d_channel_fsm;
00145
00146     Acq_Conf_Fpga* d_acq_parameters;
00147
00148     Gnss_Synchro* d_gnss_synchro;
00149
00150     uint64_t d_sample_counter;
00151
00152     const float d_threshold;
00153     float d_mag;
00154     float d_input_power;
00155     float d_test_statistics;
00156     float d_doppler_step2;
00157     float d_doppler_center_step_two;
00158
00159     int32_t d_doppler_center;
00160     int32_t d_state;
00161
00162     uint32_t d_doppler_index;
00163     uint32_t d_channel;
00164     const uint32_t d_doppler_step;
00165     const uint32_t d_doppler_max;
00166     const uint32_t d_num_doppler_bins;
00167     uint32_t d_total_block_exp;
00168     uint32_t d_num_doppler_bins_step2;
00169     uint32_t d_max_num_acqs;
00170
00171     bool d_active;
00172     bool d_make_2_steps;
00173 };
00174
00175
00176 /** \} */
00177 /** \} */
00178 #endif  // GNSS_SDR_PCPS_ACQUISITION_FPGA_H
```

13.18 pcps_assisted_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition with assistance and multi-dwells.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class pcps_assisted_acquisition_cc

This class implements a Parallel Code Phase Search Acquisition.

Typedefs

- using pcps_assisted_acquisition_cc_sptr = gnss_shared_ptr<pcps_assisted_acquisition_cc>

Functions

- pcps_assisted_acquisition_cc_sptr **pcps_make_assisted_acquisition_cc** (const Acq_Conf &conf)

13.18.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with assistance and multi-dwells.
Acquisition strategy (Kay Borre book + CFAR threshold).

1. Compute the input signal power estimation
2. Doppler serial search loop
3. Perform the FFT-based circular convolution (parallel time search)
4. Record the maximum peak and the associated synchronization parameters
5. Compute the test statistics and compare to the threshold
6. Declare positive or negative acquisition using a message queue

Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkhauser, 2007. pp 81-84

Authors

- Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_assisted_acquisition_cc.h.

13.19 pcps_assisted_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_assisted_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition with assistance and
      multi-dwells
00004  *
00005  *  Acquisition strategy (Kay Borre book + CFAR threshold).
00006  *  <ol>
00007  *  <li> Compute the input signal power estimation
00008  *  <li> Doppler serial search loop
00009  *  <li> Perform the FFT-based circular convolution (parallel time search)
00010  *  <li> Record the maximum peak and the associated synchronization parameters
00011  *  <li> Compute the test statistics and compare to the threshold
00012  *  <li> Declare positive or negative acquisition using a message queue
00013  *  </ol>
00014  *
00015  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00016  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00017  * Approach", Birkhauser, 2007. pp 81-84
00018  *
00019  * \authors <ul>
00020  *          <li> Javier Arribas, 2013. jarribas(at)cttc.es
00021  *          </ul>
00022  *
00023  * -----------------------------------------------------------------------------
00024  *
00025  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00026  * This file is part of GNSS-SDR.
00027  *
00028  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00029  * SPDX-License-Identifier: GPL-3.0-or-later
00030  *
```

```
00031  * -----------------------------------------------------------------------------
00032  */
00033
00034 #ifndef GNSS_SDR_PCPS_ASSISTED_ACQUISITION_CC_H
00035 #define GNSS_SDR_PCPS_ASSISTED_ACQUISITION_CC_H
00036
00037 #include "acq_conf.h"
00038 #include "acquisition_impl_interface.h"
00039 #include "channel_fsm.h"
00040 #include "gnss_sdr_fft.h"
00041 #include "gnss_synchro.h"
00042 #include <gnuradio/block.h>
00043 #include <gnuradio/gr_complex.h>
00044 #include <fstream>
00045 #include <memory>
00046 #include <string>
00047 #include <utility>
00048 #include <vector>
00049
00050 /** \addtogroup Acquisition
00051  * \{ */
00052 /** \addtogroup Acq_gnuradio_blocks
00053  * \{ */
00054
00055
00056 class pcps_assisted_acquisition_cc;
00057
00058 using pcps_assisted_acquisition_cc_sptr = gnss_shared_ptr<pcps_assisted_acquisition_cc>;
00059
00060 pcps_assisted_acquisition_cc_sptr pcps_make_assisted_acquisition_cc(const Acq_Conf& conf);
00061
00062 /*!
00063  * \brief This class implements a Parallel Code Phase Search Acquisition.
00064  *
00065  * Check \ref Navitec2012 "An Open Source Galileo E1 Software Receiver",
00066  * Algorithm 1, for a pseudocode description of this implementation.
00067  */
00068 class pcps_assisted_acquisition_cc : public acquisition_impl_interface
00069 {
00070 public:
00071     /*!
00072      * \brief Default destructor.
00073      */
00074     ~pcps_assisted_acquisition_cc();
00075
00076     /*!
00077      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00078      * to exchange synchronization data between acquisition and tracking blocks.
00079      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00080      */
00081     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00082     {
00083         d_gnss_synchro = p_gnss_synchro;
00084     }
00085
00086     /*!
00087      * \brief Returns the maximum peak of grid search.
00088      */
00089     inline uint32_t mag() const override
00090     {
00091         return d_test_statistics;
00092     }
00093
00094     /*!
00095      * \brief Sets local code for PCPS acquisition algorithm.
00096      * \param code - Pointer to the PRN code.
00097      */
00098     void set_local_code(std::complex<float>* code) override;
00099
00100     /*!
00101      * \brief Starts acquisition algorithm, turning from standby mode to
00102      * active mode
00103      * \param active - bool that activates/deactivates the block.
00104      */
00105     inline void set_active(bool active) override
00106     {
00107         if (!active)
00108             {
00109                 d_state = 0;
00110             }
00111
00112         d_active = active;
00113     }
00114
00115     /*!
00116      * \brief Set acquisition channel unique ID
00117      * \param channel - receiver channel.
```

```
00118      */
00119     inline void set_channel(uint32_t channel) override
00120     {
00121         d_channel = channel;
00122     }
00123
00124     /*!
00125      * \brief Set channel fsm associated to this acquisition instance
00126      */
00127     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00128     {
00129         d_channel_fsm = std::move(channel_fsm);
00130     }
00131
00132     /*!
00133      * \brief Parallel Code Phase Search Acquisition signal processing.
00134      */
00135     int general_work(int noutput_items, gr_vector_int& ninput_items,
00136         gr_vector_const_void_star& input_items,
00137         gr_vector_void_star& output_items) override;
00138
00139
00140 private:
00141     void forecast(int noutput_items, gr_vector_int& ninput_items_required) override;
00142
00143     friend pcps_assisted_acquisition_cc_sptr
00144     pcps_make_assisted_acquisition_cc(const Acq_Conf& conf);
00145
00146     explicit pcps_assisted_acquisition_cc(const Acq_Conf& conf);
00147
00148     void calculate_magnitudes(gr_complex* fft_begin, int32_t doppler_shift, int32_t doppler_offset);
00149
00150     int32_t compute_and_accumulate_grid(gr_vector_const_void_star& input_items);
00151     float estimate_input_power(gr_vector_const_void_star& input_items);
00152     float search_maximum();
00153     void get_assistance();
00154     void reset_grid();
00155     void redefine_grid();
00156
00157     std::string d_satellite_str;
00158     const Acq_Conf d_acq_params;
00159
00160     std::ofstream d_dump_file;
00161
00162     Gnss_Synchro* d_gnss_synchro;
00163
00164     uint64_t d_sample_counter;
00165
00166     float d_input_power;
00167     float d_test_statistics;
00168
00169     uint32_t d_channel;
00170     uint32_t d_code_phase;
00171     const uint32_t d_fft_size;
00172
00173     const int32_t d_gnuradio_forecast_samples;
00174     int32_t d_doppler_max;
00175     int32_t d_doppler_min;
00176     int32_t d_num_doppler_points;
00177     int32_t d_state;
00178     int32_t d_well_count;
00179
00180     bool d_active;
00181     bool d_disable_assist;
00182
00183     std::weak_ptr<ChannelFsm> d_channel_fsm;
00184     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00185     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00186
00187     std::vector<std::vector<std::complex<float>>> d_grid_doppler_wipeoffs;
00188     std::vector<std::vector<float>> d_grid_data;
00189     std::vector<gr_complex> d_fft_codes;
00190     std::vector<float> d_magnitude;
00191 };
00192
00193
00194 /** \} */
00195 /** \} */
00196 #endif  // GNSS_SDR_PCPS_ASSISTED_ACQUISITION_CC_H
```

13.20 pcps_cccwsr_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search acquisition with Coherent Channel Combining With Sign Recovery scheme.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class pcps_cccwsr_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with Coherent Channel Combining With Sign Recovery scheme.

Typedefs

- using pcps_cccwsr_acquisition_cc_sptr = gnss_shared_ptr<pcps_cccwsr_acquisition_cc>

Functions

- pcps_cccwsr_acquisition_cc_sptr **pcps_cccwsr_make_acquisition_cc** (const Acq_Conf &conf)

13.20.1 Detailed Description

This class implements a Parallel Code Phase Search acquisition with Coherent Channel Combining With Sign Recovery scheme.

Author

Marc Molina, 2013. marc.molina.pena(at)gmail.com

D.Borio, C.O'Driscoll, G.Lachapelle, "Coherent, Noncoherent and Differentially Coherent Combining Techniques for Acquisition of New Composite GNSS Signals", IEEE Transactions On Aerospace and Electronic Systems vol. 45 no. 3, July 2009, section IV

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pcps_cccwsr_acquisition_cc.h.

13.21 pcps_cccwsr_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_cccwsr_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search acquisition
00004  *  with Coherent Channel Combining With Sign Recovery scheme.
00005  * \author Marc Molina, 2013. marc.molina.pena(at)gmail.com
00006  *
00007  * D.Borio, C.O'Driscoll, G.Lachapelle, "Coherent, Noncoherent and
00008  * Differentially Coherent Combining Techniques for Acquisition of
00009  * New Composite GNSS Signals", IEEE Transactions On Aerospace and
00010  * Electronic Systems vol. 45 no. 3, July 2009, section IV
00011  *
00012  * -----------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
```

```
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_PCPS_CCCWSR_ACQUISITION_CC_H
00024 #define GNSS_SDR_PCPS_CCCWSR_ACQUISITION_CC_H
00025
00026 #include "acq_conf.h"
00027 #include "acquisition_impl_interface.h"
00028 #include "channel_fsm.h"
00029 #include "gnss_sdr_fft.h"
00030 #include "gnss_synchro.h"
00031 #include <gnuradio/block.h>
00032 #include <gnuradio/gr_complex.h>
00033 #include <fstream>
00034 #include <memory>
00035 #include <string>
00036 #include <utility>
00037 #include <vector>
00038
00039 /** \addtogroup Acquisition
00040  * \{ */
00041 /** \addtogroup Acq_gnuradio_blocks
00042  * \{ */
00043
00044
00045 class pcps_cccwsr_acquisition_cc;
00046
00047 using pcps_cccwsr_acquisition_cc_sptr = gnss_shared_ptr<pcps_cccwsr_acquisition_cc>;
00048
00049 pcps_cccwsr_acquisition_cc_sptr pcps_cccwsr_make_acquisition_cc(const Acq_Conf& conf);
00050
00051 /*!
00052  * \brief This class implements a Parallel Code Phase Search Acquisition with
00053  * Coherent Channel Combining With Sign Recovery scheme.
00054  */
00055 class pcps_cccwsr_acquisition_cc : public acquisition_impl_interface
00056 {
00057 public:
00058     /*!
00059      * \brief Default destructor.
00060      */
00061     ~pcps_cccwsr_acquisition_cc();
00062
00063     /*!
00064      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00065      * to exchange synchronization data between acquisition and tracking blocks.
00066      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00067      */
00068     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00069     {
00070         d_gnss_synchro = p_gnss_synchro;
00071     }
00072
00073     /*!
00074      * \brief Returns the maximum peak of grid search.
00075      */
00076     inline uint32_t mag() const override
00077     {
00078         return d_mag;
00079     }
00080
00081     /*!
00082      * \brief Sets local code for CCCWSR acquisition algorithm.
00083      * \param data_code - Pointer to the data PRN code.
00084      * \param pilot_code - Pointer to the pilot PRN code.
00085      */
00086     void set_local_code(std::complex<float>* code_data, std::complex<float>* code_pilot) override;
00087
00088     /*!
00089      * \brief Starts acquisition algorithm, turning from standby mode to
00090      * active mode
00091      * \param active - bool that activates/deactivates the block.
00092      */
00093     inline void set_active(bool active) override
00094     {
00095         if (!active)
00096             {
00097                 d_state = 0;
00098             }
00099
00100         d_active = active;
00101     }
00102
00103     /*!
00104      * \brief Set acquisition channel unique ID
00105      * \param channel - receiver channel.
```

```
00106      */
00107     inline void set_channel(uint32_t channel) override
00108     {
00109         d_channel = channel;
00110     }
00111
00112     /*!
00113      * \brief Set channel fsm associated to this acquisition instance
00114      */
00115     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00116     {
00117         d_channel_fsm = std::move(channel_fsm);
00118     }
00119
00120     /*!
00121      * \brief Coherent Channel Combining With Sign Recovery Acquisition signal processing.
00122      */
00123     int general_work(int noutput_items, gr_vector_int& ninput_items,
00124         gr_vector_const_void_star& input_items,
00125         gr_vector_void_star& output_items) override;
00126
00127 private:
00128     friend pcps_cccwsr_acquisition_cc_sptr
00129     pcps_cccwsr_make_acquisition_cc(const Acq_Conf& conf);
00130
00131     explicit pcps_cccwsr_acquisition_cc(const Acq_Conf& conf);
00132
00133     void calculate_magnitudes(gr_complex* fft_begin, int32_t doppler_shift, int32_t doppler_offset);
00134
00135     std::ofstream d_dump_file;
00136     std::string d_satellite_str;
00137     const Acq_Conf d_acq_params;
00138
00139     Gnss_Synchro* d_gnss_synchro;
00140
00141     int64_t d_fs_in;
00142     uint64_t d_sample_counter;
00143
00144     float d_mag;
00145     float d_input_power;
00146     float d_test_statistics;
00147
00148     int32_t d_state;
00149     uint32_t d_doppler_resolution;
00150     uint32_t d_well_count;
00151     const uint32_t d_fft_size;
00152     uint32_t d_num_doppler_bins;
00153     uint32_t d_code_phase;
00154     uint32_t d_channel;
00155
00156     bool d_active;
00157
00158     std::weak_ptr<ChannelFsm> d_channel_fsm;
00159
00160     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00161     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00162
00163     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00164     std::vector<gr_complex> d_fft_code_data;
00165     std::vector<gr_complex> d_fft_code_pilot;
00166     std::vector<gr_complex> d_data_correlation;
00167     std::vector<gr_complex> d_pilot_correlation;
00168     std::vector<gr_complex> d_correlation_plus;
00169     std::vector<gr_complex> d_correlation_minus;
00170     std::vector<float> d_magnitude;
00171 };
00172
00173
00174 /** \} */
00175 /** \} */
00176 #endif  // GNSS_SDR_PCPS_CCCWSR_ACQUISITION_CC_H
```

13.22 pcps_opencl_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition using OpenCL to offload some functions to the GPU.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_block_interface.h"
#include "gnss_sdr_fft.h"
```

```
#include "gnss_synchro.h"
#include "opencl/fft_internal.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include "opencl/cl.hpp"
#include <cstdint>
#include <fstream>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class pcps_opencl_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition.

Typedefs

- using pcps_opencl_acquisition_cc_sptr = gnss_shared_ptr<pcps_opencl_acquisition_cc>

Functions

- pcps_opencl_acquisition_cc_sptr **pcps_make_opencl_acquisition_cc** (const Acq_Conf &conf, uint32_t max_dwells)

13.22.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition using OpenCL to offload some functions to the GPU.
Acquisition strategy (Kay Borre book + CFAR threshold).

1. Compute the input signal power estimation
2. Doppler serial search loop
3. Perform the FFT-based circular convolution (parallel time search)
4. Record the maximum peak and the associated synchronization parameters
5. Compute the test statistics and compare to the threshold
6. Declare positive or negative acquisition using a message port

Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkhauser, 2007. pp 81-84

Authors

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com
- Marc Molina, 2013. marc.molina.pena@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pcps_opencl_acquisition_cc.h.

13.22.2 Macro Definition Documentation

13.22.2.1 CL_SILENCE_DEPRECATION

```
#define CL_SILENCE_DEPRECATION
```

Definition at line 40 of file pcps_opencl_acquisition_cc.h.

13.23 pcps_opencl_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_opencl_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition
00004  * using OpenCL to offload some functions to the GPU.
00005  *
00006  *  Acquisition strategy (Kay Borre book + CFAR threshold).
00007  *  <ol>
00008  *  <li> Compute the input signal power estimation
00009  *  <li> Doppler serial search loop
00010  *  <li> Perform the FFT-based circular convolution (parallel time search)
00011  *  <li> Record the maximum peak and the associated synchronization parameters
00012  *  <li> Compute the test statistics and compare to the threshold
00013  *  <li> Declare positive or negative acquisition using a message port
00014  *  </ol>
00015  *
00016  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00017  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00018  * Approach", Birkhauser, 2007. pp 81-84
00019  *
00020  * \authors <ul>
00021  *          <li> Javier Arribas, 2011. jarribas(at)cttc.es
00022  *          <li> Luis Esteve, 2012. luis(at)epsilon-formacion.com
00023  *          <li> Marc Molina, 2013. marc.molina.pena@gmail.com
00024  *          </ul>
00025  *
00026  * -----------------------------------------------------------------------------
00027  *
00028  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00029  * This file is part of GNSS-SDR.
00030  *
00031  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00032  * SPDX-License-Identifier: GPL-3.0-or-later
00033  *
00034  * -----------------------------------------------------------------------------
00035  */
00036
00037 #ifndef GNSS_SDR_PCPS_OPENCL_ACQUISITION_CC_H
00038 #define GNSS_SDR_PCPS_OPENCL_ACQUISITION_CC_H
00039
00040 #define CL_SILENCE_DEPRECATION
00041 #include "acq_conf.h"
00042 #include "acquisition_impl_interface.h"
00043 #include "channel_fsm.h"
00044 #include "gnss_block_interface.h"
00045 #include "gnss_sdr_fft.h"
00046 #include "gnss_synchro.h"
00047 #include "opencl/fft_internal.h"
00048 #include <gnuradio/block.h>
00049 #include <gnuradio/gr_complex.h>
00050 #include "opencl/cl.hpp"
00051 #include <cstdint>
00052 #include <fstream>
00053 #include <memory>  // for weak_ptr
00054 #include <string>
00055 #include <vector>
00056
00057 /** \addtogroup Acquisition
00058  * \{ */
00059 /** \addtogroup Acq_gnuradio_blocks
00060  * \{ */
00061
00062
00063 class pcps_opencl_acquisition_cc;
00064
00065 using pcps_opencl_acquisition_cc_sptr = gnss_shared_ptr<pcps_opencl_acquisition_cc>;
00066
00067 pcps_opencl_acquisition_cc_sptr pcps_make_opencl_acquisition_cc(const Acq_Conf& conf, uint32_t
      max_dwells);
00068
00069 /*!
00070  * \brief This class implements a Parallel Code Phase Search Acquisition.
00071  *
00072  * Check \ref Navitec2012 "An Open Source Galileo E1 Software Receiver",
00073  * Algorithm 1, for a pseudocode description of this implementation.
00074  */
00075 class pcps_opencl_acquisition_cc : public acquisition_impl_interface
00076 {
00077 public:
00078     /*!
00079      * \brief Default destructor.
00080      */
00081     ~pcps_opencl_acquisition_cc();
00082
```

```
00083     /*!
00084      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00085      * to exchange synchronization data between acquisition and tracking blocks.
00086      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00087      */
00088     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00089     {
00090         d_gnss_synchro = p_gnss_synchro;
00091     }
00092
00093     /*!
00094      * \brief Returns the maximum peak of grid search.
00095      */
00096     inline uint32_t mag() const override
00097     {
00098         return d_mag;
00099     }
00100
00101     /*!
00102      * \brief Sets local code for PCPS acquisition algorithm.
00103      * \param code - Pointer to the PRN code.
00104      */
00105     void set_local_code(std::complex<float>* code) override;
00106
00107     /*!
00108      * \brief Starts acquisition algorithm, turning from standby mode to
00109      * active mode
00110      * \param active - bool that activates/deactivates the block.
00111      */
00112     inline void set_active(bool active) override
00113     {
00114         if (!active)
00115             {
00116                 d_state = 0;
00117             }
00118
00119         d_active = active;
00120     }
00121
00122     /*!
00123      * \brief Set acquisition channel unique ID
00124      * \param channel - receiver channel.
00125      */
00126     inline void set_channel(uint32_t channel) override
00127     {
00128         d_channel = channel;
00129     }
00130
00131     /*!
00132      * \brief Set channel fsm associated to this acquisition instance
00133      */
00134     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00135     {
00136         d_channel_fsm = channel_fsm;
00137     }
00138
00139     inline bool opencl_ready() const
00140     {
00141         bool ready = false;
00142         if (d_opencl == 0)
00143             {
00144                 ready = true;
00145             }
00146         return ready;
00147     }
00148
00149     void acquisition_core_volk();
00150
00151     void acquisition_core_opencl();
00152
00153     /*!
00154      * \brief Parallel Code Phase Search Acquisition signal processing.
00155      */
00156     int general_work(int noutput_items, gr_vector_int& ninput_items,
00157         gr_vector_const_void_star& input_items,
00158         gr_vector_void_star& output_items) override;
00159
00160 private:
00161     friend pcps_opencl_acquisition_cc_sptr
00162     pcps_make_opencl_acquisition_cc(const Acq_Conf& conf, uint32_t max_dwells);
00163
00164     explicit pcps_opencl_acquisition_cc(const Acq_Conf& conf, uint32_t max_dwells);
00165
00166     void calculate_magnitudes(gr_complex* fft_begin, int doppler_shift, int doppler_offset);
00167
00168     int init_opencl_environment(const std::string& kernel_filename);
00169
```

```
00170     cl::Platform d_cl_platform;
00171     cl::Device d_cl_device;
00172     cl::Context d_cl_context;
00173     cl::Program d_cl_program;
00174     cl::Buffer* d_cl_buffer_in;
00175     cl::Buffer* d_cl_buffer_fft_codes;
00176     cl::Buffer* d_cl_buffer_1;
00177     cl::Buffer* d_cl_buffer_2;
00178     cl::Buffer* d_cl_buffer_magnitude;
00179     cl::Buffer** d_cl_buffer_grid_doppler_wipeoffs;
00180     cl::CommandQueue* d_cl_queue;
00181     clFFT_Plan d_cl_fft_plan;
00182     cl_int d_cl_fft_batch_size;
00183
00184     std::string d_satellite_str;
00185     const Acq_Conf d_acq_params;
00186
00187     std::ofstream d_dump_file;
00188
00189     Gnss_Synchro* d_gnss_synchro;
00190
00191     uint64_t d_sample_counter;
00192
00193     int* d_max_doppler_indexs;
00194
00195     float d_mag;
00196     float d_input_power;
00197     float d_test_statistics;
00198
00199     int d_state;
00200     int d_opencl;
00201
00202     uint32_t d_max_dwells;
00203     uint32_t d_well_count;
00204     const uint32_t d_fft_size;
00205     uint32_t d_fft_size_pow2;
00206     uint32_t d_num_doppler_bins;
00207     uint32_t d_code_phase;
00208     uint32_t d_channel;
00209     uint32_t d_in_dwell_count;
00210
00211     bool d_active;
00212     bool d_core_working;
00213
00214     std::weak_ptr<ChannelFsm> d_channel_fsm;
00215
00216     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00217     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00218
00219     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00220     std::vector<std::vector<gr_complex» d_in_buffer;
00221     std::vector<gr_complex> d_fft_codes;
00222     std::vector<gr_complex> d_zero_vector;
00223     std::vector<uint64_t> d_sample_counter_buffer;
00224     std::vector<float> d_magnitude;
00225 };
00226
00227
00228 /** \} */
00229 /** \} */
00230 #endif  // GNSS_SDR_PCPS_OPENCL_ACQUISITION_CC_H
```

13.24 pcps_quicksync_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition with the QuickSync Algorithm.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <cassert>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class pcps_quicksync_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with the implementation of the Sparse QuickSync Algorithm.

Typedefs

- using pcps_quicksync_acquisition_cc_sptr = gnss_shared_ptr<pcps_quicksync_acquisition_cc>

Functions

- pcps_quicksync_acquisition_cc_sptr **pcps_quicksync_make_acquisition_cc** (const Acq_Conf &conf, uint32_t folding_factor, uint32_t max_dwells)

13.24.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with the QuickSync Algorithm.
Acquisition strategy (Kay Borre book CFAR + threshold).

1. Compute the input signal power estimation
2. Doppler serial search loop
3. Perform folding of the incoming signal and local generated code
4. Perform the FFT-based circular convolution (parallel time search)
5. Record the maximum peak and the associated synchronization parameters
6. Compute the test statistics and compare to the threshold
7. Declare positive or negative acquisition using a message port
8. Obtain the adequate acquisition parameters by correlating the incoming signal shifted by the possible folded delays

Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, "A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach", Birkha user, 2007. pp 81-84

Date

Jun2 2014

Author

Damian Miralles Sanchez, dmiralles2009@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pcps_quicksync_acquisition_cc.h.

13.25 pcps_quicksync_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_quicksync_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition with the
00004  * QuickSync Algorithm
00005  *
00006  *  Acquisition strategy (Kay Borre book CFAR + threshold).
00007  *  <ol>
00008  *  <li> Compute the input signal power estimation
00009  *  <li> Doppler serial search loop
```

```
00010  *  <li> Perform folding of the incoming signal and local generated code
00011  *  <li> Perform the FFT-based circular convolution (parallel time search)
00012  *  <li> Record the maximum peak and the associated synchronization parameters
00013  *  <li> Compute the test statistics and compare to the threshold
00014  *  <li> Declare positive or negative acquisition using a message port
00015  *  <li> Obtain the adequate acquisition parameters by correlating the incoming
00016  *       signal shifted by the possible folded delays
00017  *  </ol>
00018  *
00019  * Kay Borre book: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00020  * "A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00021  * Approach", Birkha user, 2007. pp 81-84
00022  *
00023  * \date Jun2 2014
00024  * \author Damian Miralles Sanchez, dmiralles2009@gmail.com
00025  *
00026  * -----------------------------------------------------------------------------
00027  *
00028  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00029  * This file is part of GNSS-SDR.
00030  *
00031  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00032  * SPDX-License-Identifier: GPL-3.0-or-later
00033  *
00034  * -----------------------------------------------------------------------------
00035  */
00036
00037 #ifndef GNSS_SDR_PCPS_QUICKSYNC_ACQUISITION_CC_H
00038 #define GNSS_SDR_PCPS_QUICKSYNC_ACQUISITION_CC_H
00039
00040 #include "acq_conf.h"
00041 #include "acquisition_impl_interface.h"
00042 #include "channel_fsm.h"
00043 #include "gnss_sdr_fft.h"
00044 #include "gnss_synchro.h"
00045 #include <gnuradio/block.h>
00046 #include <gnuradio/gr_complex.h>
00047 #include <cassert>
00048 #include <fstream>
00049 #include <memory>  // for weak_ptr
00050 #include <string>
00051 #include <utility>
00052 #include <vector>
00053
00054 /** \addtogroup Acquisition
00055  * \{ */
00056 /** \addtogroup Acq_gnuradio_blocks
00057  * \{ */
00058
00059
00060 class pcps_quicksync_acquisition_cc;
00061
00062 using pcps_quicksync_acquisition_cc_sptr = gnss_shared_ptr<pcps_quicksync_acquisition_cc>;
00063
00064 pcps_quicksync_acquisition_cc_sptr pcps_quicksync_make_acquisition_cc(const Acq_Conf& conf, uint32_t
      folding_factor, uint32_t max_dwells);
00065
00066 /*!
00067  * \brief This class implements a Parallel Code Phase Search Acquisition with
00068  * the implementation of the Sparse QuickSync Algorithm.
00069  *
00070  * Check \ref Navitec2012 "Faster GPS via the Sparse Fourier Transform",
00071  * for details of its implementation and functionality.
00072  */
00073 class pcps_quicksync_acquisition_cc : public acquisition_impl_interface
00074 {
00075 public:
00076     /*!
00077      * \brief Default destructor.
00078      */
00079     ~pcps_quicksync_acquisition_cc();
00080
00081     /*!
00082      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00083      * to exchange synchronization data between acquisition and tracking blocks.
00084      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00085      */
00086     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00087     {
00088         d_gnss_synchro = p_gnss_synchro;
00089     }
00090
00091     /*!
00092      * \brief Returns the maximum peak of grid search.
00093      */
00094     inline uint32_t mag() const override
00095     {
```

```
00096         return d_mag;
00097     }
00098
00099     /*!
00100      * \brief Sets local code for PCPS acquisition algorithm.
00101      * \param code - Pointer to the PRN code.
00102      */
00103     void set_local_code(std::complex<float>* code) override;
00104
00105     /*!
00106      * \brief Starts acquisition algorithm, turning from standby mode to
00107      * active mode
00108      * \param active - bool that activates/deactivates the block.
00109      */
00110     inline void set_active(bool active) override
00111     {
00112         if (!active)
00113             {
00114                 d_state = 0;
00115             }
00116
00117         d_active = active;
00118     }
00119
00120     /*!
00121      * \brief Set acquisition channel unique ID
00122      * \param channel - receiver channel.
00123      */
00124     inline void set_channel(uint32_t channel) override
00125     {
00126         d_channel = channel;
00127     }
00128
00129     /*!
00130      * \brief Set channel fsm associated to this acquisition instance
00131      */
00132     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00133     {
00134         d_channel_fsm = std::move(channel_fsm);
00135     }
00136
00137     /*!
00138      * \brief Parallel Code Phase Search Acquisition signal processing.
00139      */
00140     int general_work(int noutput_items, gr_vector_int& ninput_items,
00141         gr_vector_const_void_star& input_items,
00142         gr_vector_void_star& output_items) override;
00143
00144 private:
00145     friend pcps_quicksync_acquisition_cc_sptr
00146     pcps_quicksync_make_acquisition_cc(const Acq_Conf& conf, uint32_t folding_factor, uint32_t
      max_dwells);
00147
00148     explicit pcps_quicksync_acquisition_cc(const Acq_Conf& conf, uint32_t folding_factor, uint32_t
      max_dwells);
00149
00150     void calculate_magnitudes(gr_complex* fft_begin, int32_t doppler_shift, int32_t doppler_offset);
00151
00152     std::string d_satellite_str;
00153     const Acq_Conf d_acq_params;
00154
00155     std::ofstream d_dump_file;
00156
00157     Gnss_Synchro* d_gnss_synchro;
00158
00159     uint64_t d_sample_counter;
00160
00161     float d_noise_floor_power;
00162     float d_mag;
00163     float d_input_power;
00164     float d_test_statistics;
00165     const int32_t d_vector_length;
00166     const int32_t d_samples_per_code;
00167     int32_t d_state;
00168     uint32_t d_channel;
00169     const uint32_t d_folding_factor;  // also referred in the paper as 'p'
00170     const uint32_t d_max_dwells;
00171     uint32_t d_well_count;
00172     const uint32_t d_fft_size;
00173     uint32_t d_num_doppler_bins;
00174     uint32_t d_code_phase;
00175
00176     bool d_active;
00177
00178     std::weak_ptr<ChannelFsm> d_channel_fsm;
00179
00180     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
```

```
00181     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00182
00183     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00184     std::vector<gr_complex> d_code;
00185     std::vector<gr_complex> d_fft_codes;
00186     std::vector<gr_complex> d_code_folded;
00187     std::vector<gr_complex> d_in_temp;
00188     std::vector<gr_complex> d_in_lcode;
00189     std::vector<gr_complex> d_corr_output;
00190     std::vector<float> d_magnitude;
00191     std::vector<float> d_corr_output_f;
00192     std::vector<float> d_magnitude_folded;
00193     std::vector<uint32_t> d_possible_delay;
00194 };
00195
00196
00197 /** \} */
00198 /** \} */
00199 #endif  // GNSS_SDR_PCPS_QUICKSYNC_ACQUISITION_CC_H
```

13.26 pcps_tong_acquisition_cc.h File Reference

This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.

```
#include "acq_conf.h"
#include "acquisition_impl_interface.h"
#include "channel_fsm.h"
#include "gnss_sdr_fft.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class pcps_tong_acquisition_cc

  This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.

Typedefs

- using pcps_tong_acquisition_cc_sptr = gnss_shared_ptr<pcps_tong_acquisition_cc>

Functions

- pcps_tong_acquisition_cc_sptr **pcps_tong_make_acquisition_cc** (const Acq_Conf &conf, uint32_t tong↩ _init_val, uint32_t tong_max_val, uint32_t tong_max_dwells)

13.26.1 Detailed Description

This class implements a Parallel Code Phase Search Acquisition with Tong algorithm.

Author

Marc Molina, 2013. marc.molina.pena(at)gmail.com

Acquisition strategy (Kaplan book + CFAR threshold).

1. Compute the input signal power estimation.
2. Doppler serial search loop.
3. Perform the FFT-based circular convolution (parallel time search).

4. Compute the tests statistics for all the cells.
5. Accumulate the grid of tests statistics with the previous grids.
6. Record the maximum peak and the associated synchronization parameters.
7. Compare the maximum averaged test statistics with a threshold.
8. If the test statistics exceeds the threshold, increment the Tong counter.
9. Otherwise, decrement the Tong counter.
10. If the Tong counter is equal to a given maximum value, declare positive
11. acquisition. If the Tong counter is equa to zero, declare negative
12. acquisition. Otherwise, process the next block.

Kaplan book: D.Kaplan, J.Hegarty, "Understanding GPS. Principles and Applications", Artech House, 2006, pp 223-227

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pcps_tong_acquisition_cc.h.

13.27 pcps_tong_acquisition_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pcps_tong_acquisition_cc.h
00003  * \brief This class implements a Parallel Code Phase Search Acquisition with
00004  * Tong algorithm.
00005  * \author Marc Molina, 2013. marc.molina.pena(at)gmail.com
00006  *
00007  *  Acquisition strategy (Kaplan book + CFAR threshold).
00008  *  <ol>
00009  *  <li> Compute the input signal power estimation.
00010  *  <li> Doppler serial search loop.
00011  *  <li> Perform the FFT-based circular convolution (parallel time search).
00012  *  <li> Compute the tests statistics for all the cells.
00013  *  <li> Accumulate the grid of tests statistics with the previous grids.
00014  *  <li> Record the maximum peak and the associated synchronization parameters.
00015  *  <li> Compare the maximum averaged test statistics with a threshold.
00016  *  <li> If the test statistics exceeds the threshold, increment the Tong counter.
00017  *  <li>   Otherwise, decrement the Tong counter.
00018  *  <li> If the Tong counter is equal to a given maximum value, declare positive
00019  *  <li>   acquisition. If the Tong counter is equa to zero, declare negative
00020  *  <li>   acquisition. Otherwise, process the next block.
00021  *  </ol>
00022  *
00023  * Kaplan book: D.Kaplan, J.Hegarty, "Understanding GPS. Principles
00024  * and Applications", Artech House, 2006, pp 223-227
00025  *
00026  * -----------------------------------------------------------------------------
00027  *
00028  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00029  * This file is part of GNSS-SDR.
00030  *
00031  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00032  * SPDX-License-Identifier: GPL-3.0-or-later
00033  *
00034  * -----------------------------------------------------------------------------
00035  */
00036
00037 #ifndef GNSS_SDR_PCPS_TONG_ACQUISITION_CC_H
00038 #define GNSS_SDR_PCPS_TONG_ACQUISITION_CC_H
00039
00040 #include "acq_conf.h"
00041 #include "acquisition_impl_interface.h"
00042 #include "channel_fsm.h"
00043 #include "gnss_sdr_fft.h"
00044 #include "gnss_synchro.h"
00045 #include <gnuradio/block.h>
00046 #include <gnuradio/gr_complex.h>
00047 #include <fstream>
00048 #include <memory>  // for weak_ptr
00049 #include <string>
00050 #include <utility>
00051 #include <vector>
```

```
00052
00053 /** \addtogroup Acquisition
00054  * \{ */
00055 /** \addtogroup Acq_gnuradio_blocks
00056  * \{ */
00057
00058
00059 class pcps_tong_acquisition_cc;
00060
00061 using pcps_tong_acquisition_cc_sptr = gnss_shared_ptr<pcps_tong_acquisition_cc>;
00062
00063 pcps_tong_acquisition_cc_sptr pcps_tong_make_acquisition_cc(
00064     const Acq_Conf& conf,
00065     uint32_t tong_init_val,
00066     uint32_t tong_max_val,
00067     uint32_t tong_max_dwells);
00068
00069 /*!
00070  * \brief This class implements a Parallel Code Phase Search Acquisition with
00071  * Tong algorithm.
00072  */
00073 class pcps_tong_acquisition_cc : public acquisition_impl_interface
00074 {
00075 public:
00076     /*!
00077      * \brief Default destructor.
00078      */
00079     ~pcps_tong_acquisition_cc();
00080
00081     /*!
00082      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00083      * to exchange synchronization data between acquisition and tracking blocks.
00084      * \param p_gnss_synchro Satellite information shared by the processing blocks.
00085      */
00086     inline void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override
00087     {
00088         d_gnss_synchro = p_gnss_synchro;
00089     }
00090
00091     /*!
00092      * \brief Returns the maximum peak of grid search.
00093      */
00094     inline uint32_t mag() const override
00095     {
00096         return d_mag;
00097     }
00098
00099     /*!
00100      * \brief Sets local code for TONG acquisition algorithm.
00101      * \param code - Pointer to the PRN code.
00102      */
00103     void set_local_code(std::complex<float>* code) override;
00104
00105     /*!
00106      * \brief Starts acquisition algorithm, turning from standby mode to
00107      * active mode
00108      * \param active - bool that activates/deactivates the block.
00109      */
00110     inline void set_active(bool active) override
00111     {
00112         if (!active)
00113             {
00114                 d_state = 0;
00115             }
00116
00117         d_active = active;
00118     }
00119
00120     /*!
00121      * \brief Set acquisition channel unique ID
00122      * \param channel - receiver channel.
00123      */
00124     inline void set_channel(uint32_t channel) override
00125     {
00126         d_channel = channel;
00127     }
00128
00129     /*!
00130      * \brief Set channel fsm associated to this acquisition instance
00131      */
00132     inline void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) override
00133     {
00134         d_channel_fsm = std::move(channel_fsm);
00135     }
00136
00137     /*!
00138      * \brief Parallel Code Phase Search Acquisition signal processing.
```

```
00139      */
00140     int general_work(int noutput_items, gr_vector_int& ninput_items,
00141         gr_vector_const_void_star& input_items,
00142         gr_vector_void_star& output_items) override;
00143
00144 private:
00145     friend pcps_tong_acquisition_cc_sptr
00146     pcps_tong_make_acquisition_cc(
00147         const Acq_Conf& conf,
00148         uint32_t tong_init_val,
00149         uint32_t tong_max_val,
00150         uint32_t tong_max_dwells);
00151
00152     pcps_tong_acquisition_cc(
00153         const Acq_Conf& conf,
00154         uint32_t tong_init_val,
00155         uint32_t tong_max_val,
00156         uint32_t tong_max_dwells);
00157
00158     void calculate_magnitudes(gr_complex* fft_begin, int32_t doppler_shift, int32_t doppler_offset);
00159
00160     std::string d_satellite_str;
00161     const Acq_Conf d_acq_params;
00162
00163     std::ofstream d_dump_file;
00164
00165     Gnss_Synchro* d_gnss_synchro;
00166
00167     uint64_t d_sample_counter;
00168
00169     float d_mag;
00170     float d_input_power;
00171     float d_test_statistics;
00172     int32_t d_state;
00173     uint32_t d_channel;
00174     uint32_t d_dwell_count;
00175     const uint32_t d_tong_init_val;
00176     const uint32_t d_tong_max_val;
00177     const uint32_t d_tong_max_dwells;
00178     uint32_t d_tong_count;
00179     const uint32_t d_fft_size;
00180     uint32_t d_num_doppler_bins;
00181     uint32_t d_code_phase;
00182
00183     bool d_active;
00184
00185     std::weak_ptr<ChannelFsm> d_channel_fsm;
00186     std::unique_ptr<gnss_fft_complex_fwd> d_fft_if;
00187     std::unique_ptr<gnss_fft_complex_rev> d_ifft;
00188
00189     std::vector<std::vector<gr_complex» d_grid_doppler_wipeoffs;
00190     std::vector<std::vector<float» d_grid_data;
00191     std::vector<gr_complex> d_fft_codes;
00192     std::vector<float> d_magnitude;
00193 };
00194
00195
00196 /** \} */
00197 /** \} */
00198 #endif  // GNSS_SDR_PCPS_TONG_ACQUISITION_CC_H
```

13.28 acq_conf.h File Reference

Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm.

```
#include "configuration_interface.h"
#include <gnuradio/gr_complex.h>
#include <cstdint>
#include <string>
```

Classes

- class Acq_Conf

13.28.1 Detailed Description

Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm.

Author

Carles Fernandez, 2018. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file acq_conf.h.

13.29 acq_conf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file acq_conf.h
00003  * \brief Class that contains all the configuration parameters for generic
00004  * acquisition block based on the PCPS algorithm.
00005  * \author Carles Fernandez, 2018. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_ACQ_CONF_H
00019 #define GNSS_SDR_ACQ_CONF_H
00020
00021 #include "configuration_interface.h"
00022 #include <gnuradio/gr_complex.h>
00023 #include <cstdint>
00024 #include <string>
00025
00026 /** \addtogroup Acquisition
00027  * \{ */
00028 /** \addtogroup acquisition_libs acquisition_libs
00029  * Library with utilities for GNSS signal acquisition
00030  * \{ */
00031
00032
00033 class Acq_Conf
00034 {
00035 public:
00036     Acq_Conf() = default;
00037
00038     void SetFromConfiguration(const ConfigurationInterface *configuration, const std::string &role,
      double chip_rate, double opt_freq);
00039
00040     /*!
00041      * \brief Number of samples processed per dwell: floor(sampled_ms * resampled_fs / 1000),
00042      * with the window doubled before flooring if bit_transition_flag is set.
00043      * Computed with integer arithmetic, so it is exact for any resampled_fs.
00044      */
00045     uint32_t GetSamplesPerDwell() const;
00046
00047     /*!
00048      * \brief Fraction of a sample per dwell dropped by GetSamplesPerDwell(), in [0, 1).
00049      */
00050     double GetDwellResidualSamples() const;
00051
00052     /* PCPS Acquisition configuration */
00053     std::string item_type{"gr_complex"};
00054     std::string dump_filename;
00055
00056     int64_t fs_in{4000000LL};
00057     int64_t resampled_fs{0LL};
00058
00059     size_t it_size{sizeof(gr_complex)};
00060
00061     float samples_per_ms{0.0};
00062     float doppler_step2{125.0};
00063     float threshold{0.0};
00064     float pfa{0.0};
00065     float pfa2{0.0};
00066     float samples_per_code{0.0};
00067     float resampler_ratio{1.0};
00068
00069     uint32_t sampled_ms{1U};
00070     uint32_t ms_per_code{1U};
00071     uint32_t samples_per_chip{2U};
```

```
00072     uint32_t chips_per_second{1023000U};
00073     uint32_t max_dwells{1U};
00074     uint32_t num_doppler_bins_step2{4U};
00075     uint32_t resampler_latency_samples{0U};
00076     uint32_t dump_channel{0U};
00077     int32_t doppler_max{5000};
00078     int32_t doppler_min{-5000};
00079     int32_t doppler_step{500};
00080
00081     bool bit_transition_flag{false};
00082     bool use_CFAR_algorithm_flag{true};
00083     bool dump{false};
00084     bool blocking{true};
00085     bool blocking_on_standby{false};  // enable it only for unit testing to avoid sample consume on
      idle status
00086     bool make_2_steps{false};
00087     bool use_automatic_resampler{false};
00088     bool enable_monitor_output{false};
00089     // CFAR reference separation target, in correlation sidelobes (~1/T Hz).
00090     // Determines whether a full grid needs extra reference rows. References stay
00091     // within doppler_max: the filter response beyond it can bias noise estimates.
00092     uint32_t reference_bin_min_sidelobes{4U};
00093
00094     // Opt-in accumulation through max_dwells before thresholding, allowing a
00095     // later, stronger peak to win (Acquisition_<signal>.full_grid_search).
00096     bool full_grid_search{false};
00097
00098     // Evaluate the PCPS grid on a CUDA GPU (requires ENABLE_CUDA at build time)
00099     bool use_cuda{false};
00100     int32_t cuda_device{-1};  // CUDA device ordinal, -1 = default device
00101
00102     // Specific to some implementations
00103     bool acquire_pilot{false};
00104     bool acquire_iq{false};
00105     bool cboc{false};
00106     bool qmboc{false};
00107     int zero_padding{0};
00108     uint32_t folding_factor{0};
00109
00110     // Not part of the configuration interface
00111     uint32_t num_codes{0};
00112     uint32_t code_length{0};
00113     uint32_t vector_length{0};
00114
00115 private:
00116     void SetDerivedParams();
00117
00118     // True dwell length in samples, multiplied by 1000 (exact integer).
00119     uint64_t GetDwellSamplesTimes1000() const;
00120
00121     void ConfigureAutomaticResampler(double opt_freq);
00122 };
00123
00124
00125 /** \} */
00126 /** \} */
00127 #endif  // GNSS_SDR_ACQ_CONF_H
```

13.30 acq_conf_fpga.h File Reference

Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm running in the FPGA.

```
#include "configuration_interface.h"
#include <gnuradio/gr_complex.h>
#include <cstdint>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Acq_Conf_Fpga

13.30.1 Detailed Description

Class that contains all the configuration parameters for generic acquisition block based on the PCPS algorithm running in the FPGA.

Author

Marc Majoral, 2022. mmajoral(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file acq_conf_fpga.h.

13.31 acq_conf_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file acq_conf_fpga.h
00003  * \brief Class that contains all the configuration parameters for generic
00004  * acquisition block based on the PCPS algorithm running in the FPGA.
00005  * \author Marc Majoral, 2022. mmajoral(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_ACQ_CONF_FPGA_H
00019 #define GNSS_SDR_ACQ_CONF_FPGA_H
00020
00021 #include "configuration_interface.h"
00022 #include <gnuradio/gr_complex.h>
00023 #include <cstdint>
00024 #include <string>
00025 #include <utility>  // for std::move, std::pair
00026 #include <vector>   // for std::vector
00027
00028 /** \addtogroup Acquisition
00029  * \{ */
00030 /** \addtogroup acquisition_libs acquisition_libs
00031  * Library with utilities for GNSS signal acquisition
00032  * \{ */
00033
00034
00035 class Acq_Conf_Fpga
00036 {
00037 public:
00038     Acq_Conf_Fpga() = default;
00039
00040     void SetFromConfiguration(const ConfigurationInterface *configuration, const std::string &role,
      uint32_t blk_exp, double code_chips_per_sec, double num_chips_per_code);
00041
00042     bool ConfigureAutomaticResampler(std::vector<std::pair<uint32_t, uint32_t»
     downsampling_filter_specs, uint32_t max_FFT_size, double opt_freq);
00043
00044     bool Is_acq_config_valid(uint32_t max_FFT_size) const;
00045
00046     /* PCPS Acquisition configuration */
00047     std::string device_name = "uio0";
00048     double code_rate_cps;
00049     double code_length_chips;
00050     float doppler_step{250.0};
00051     float doppler_step2{125.0};
00052     float threshold{0.0};
00053
00054     int64_t fs_in{4000000LL};
00055     int64_t resampled_fs{4000000LL};
00056
00057     uint32_t *all_fft_codes = nullptr;  // pointer to memory that contains all the code ffts
00058     uint32_t num_doppler_bins_step2{4U};
00059     uint32_t downsampling_filter_num{0U};
00060     uint32_t downsampling_factor{1U};
00061     uint32_t downsampling_filter_delay{0U};
00062     uint32_t total_block_exp{13U};
00063     uint32_t excludelimit{5U};
```

```
00064     uint32_t max_num_acqs{2U};
00065     uint32_t fft_size{1U};
00066     uint32_t code_length{16000U};
00067     int32_t doppler_max{5000};
00068
00069     bool make_2_steps{false};
00070     bool enable_zero_padding{false};
00071     bool repeat_satellite{false};
00072
00073 private:
00074     const std::string acquisition_device_name = "acquisition_S00_AXI";  // UIO device name
00075 };
00076
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_ACQ_CONF_FPGA_H
```

13.32 cuda_pcps_engine.h File Reference

GPU (CUDA + cuFFT) implementation of the Parallel Code Phase Search grid.

```
#include <complex>
#include <cstdint>
#include <memory>
#include <string>
```

Classes

- class CudaPcpsEngine

  Batched PCPS grid evaluation on a CUDA device.

13.32.1 Detailed Description

GPU (CUDA + cuFFT) implementation of the Parallel Code Phase Search grid.

Author

Phillip Vu, 2026. phillipvu(at)users.noreply.github.com

Computes, for a batch of Doppler bins, the squared magnitude of the circular cross-correlation between the carrier-wiped input and the local code, using batched FFTs on an NVIDIA GPU. It is the GPU counterpart of the inner loop of pcps_acquisition::doppler_grid(). The public interface deliberately contains no CUDA types so that it can be consumed from plain C++ translation units; the implementation lives in cuda_pcps_engine.cu.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file cuda_pcps_engine.h.

13.33 cuda_pcps_engine.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cuda_pcps_engine.h
00003  * \brief GPU (CUDA + cuFFT) implementation of the Parallel Code Phase Search grid
00004  * \author Phillip Vu, 2026. phillipvu(at)users.noreply.github.com
00005  *
00006  * Computes, for a batch of Doppler bins, the squared magnitude of the circular
00007  * cross-correlation between the carrier-wiped input and the local code, using
00008  * batched FFTs on an NVIDIA GPU. It is the GPU counterpart of the inner loop of
00009  * pcps_acquisition::doppler_grid(). The public interface deliberately contains
00010  * no CUDA types so that it can be consumed from plain C++ translation units;
00011  * the implementation lives in cuda_pcps_engine.cu.
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
```

```
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_CUDA_PCPS_ENGINE_H
00025 #define GNSS_SDR_CUDA_PCPS_ENGINE_H
00026
00027 #include <complex>
00028 #include <cstdint>
00029 #include <memory>
00030 #include <string>
00031
00032 /** \addtogroup Acquisition
00033  * \{ */
00034 /** \addtogroup acquisition_libs
00035  * \{ */
00036
00037
00038 /*!
00039  * \brief Batched PCPS grid evaluation on a CUDA device.
00040  *
00041  * Typical usage (mirrors pcps_acquisition):
00042  *   1. Construct with the FFT geometry of the acquisition block.
00043  *   2. set_doppler_wipeoffs() whenever the Doppler grid changes.
00044  *   3. set_fft_codes() whenever the local code (PRN) changes.
00045  *   4. compute_grid() once per dwell; the magnitude grid is written to the
00046  *      host rows supplied by the caller, so the existing CPU peak-search and
00047  *      statistics code can be reused unchanged.
00048  *
00049  * Each instance owns a CUDA stream, so several channels acquiring at the same
00050  * time run their grids concurrently on the device.
00051  */
00052 class CudaPcpsEngine
00053 {
00054 public:
00055     //! Identifier of the Doppler grid uploaded with set_doppler_wipeoffs()
00056     enum GridId : int
00057     {
00058         MAIN_GRID = 0,   //!< Coarse grid (doppler_min .. doppler_max, doppler_step)
00059         STEP2_GRID = 1,  //!< Fine grid used when make_two_steps=true
00060         NUM_GRIDS = 2
00061     };
00062
00063     /*!
00064      * \param fft_size            Length of the FFTs (input vector length).
00065      * \param effective_fft_size  Number of code-phase cells written per Doppler bin
00066      *                            (fft_size, or fft_size/2 when bit_transition_flag is set).
00067      * \param max_doppler_bins    Upper bound on the number of bins in any grid.
00068      * \param device              CUDA device ordinal, or -1 for the current/default device.
00069      */
00070     CudaPcpsEngine(uint32_t fft_size, uint32_t effective_fft_size, uint32_t max_doppler_bins, int
      device = -1);
00071     ~CudaPcpsEngine();
00072
00073     CudaPcpsEngine(const CudaPcpsEngine&) = delete;
00074     CudaPcpsEngine& operator=(const CudaPcpsEngine&) = delete;
00075     CudaPcpsEngine(CudaPcpsEngine&&) = delete;
00076     CudaPcpsEngine& operator=(CudaPcpsEngine&&) = delete;
00077
00078     //! True when construction succeeded and the engine can be used.
00079     bool is_valid() const;
00080
00081     //! Human-readable description of the last error (empty if none).
00082     const std::string& last_error() const;
00083
00084     //! Name of the CUDA device in use (empty if not valid).
00085     const std::string& device_name() const;
00086
00087     /*!
00088      * \brief Upload conj(FFT(local code)), fft_size elements.
00089      * The upload completes before returning; the caller may reuse fft_codes.
00090      */
00091     bool set_fft_codes(const std::complex<float>* fft_codes);
00092
00093     /*!
00094      * \brief Upload the Doppler wipe-off carriers for one grid.
00095      * Uploads complete before returning; the caller may reuse the host rows.
00096      * \param grid      MAIN_GRID or STEP2_GRID.
00097      * \param wipeoffs  Array of `bins` host pointers, each pointing at fft_size complex samples.
00098      * \param bins      Number of Doppler bins (<= max_doppler_bins).
00099      */
00100     bool set_doppler_wipeoffs(GridId grid, const std::complex<float>* const* wipeoffs, uint32_t bins);
00101
00102     /*!
00103      * \brief Evaluate the PCPS grid for `bins` Doppler bins.
00104      *
00105      * For every bin k:  mag_k[i] (+)= | IFFT( FFT(in .* w_k) .* C )[offset + i] |^2,
```

```
00106      * with C = conj(FFT(code)), i in [0, effective_fft_size).
00107      *
00108      * \param in            Input samples (fft_size complex values, host memory).
00109      * \param grid          Which wipe-off grid to use.
00110      * \param bins          Number of Doppler bins to compute (<= bins uploaded for `grid').
00111      * \param offset        First sample of the output window (fft_size/2 when bit_transition_flag).
00112      * \param accumulate    false: overwrite the output rows; true: add to their current contents
00113      *                      (non-coherent integration across dwells). Accumulation is kept on
00114      *                      the device, so the host rows are always the full running sum.
00115      * \param magnitude_out Array of `bins' host pointers, each with room for effective_fft_size
     floats.
00116      */
00117     bool compute_grid(const std::complex<float>* in, GridId grid, uint32_t bins, uint32_t offset,
00118         bool accumulate, float* const* magnitude_out);
00119
00120     //! Number of bytes allocated on the device by this instance.
00121     size_t device_bytes() const;
00122
00123 private:
00124     struct Impl;
00125     std::unique_ptr<Impl> p;
00126     bool run_pipeline(int grid, uint32_t bins, uint32_t offset, bool accumulate);
00127 };
00128
00129
00130 /** \} */
00131 /** \} */
00132 #endif  // GNSS_SDR_CUDA_PCPS_ENGINE_H
```

13.34 fpga_acquisition.h File Reference

Highly optimized FPGA vector correlator class.

```
#include <cstdint>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Fpga_Acquisition

  Class that implements carrier wipe-off and correlators.

13.34.1 Detailed Description

Highly optimized FPGA vector correlator class.

Authors

- Marc Majoral, 2019. mmajoral(at)cttc.cat

Class that controls and executes a highly optimized acquisition HW accelerator in the FPGA
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fpga_acquisition.h.

13.35 fpga_acquisition.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_acquisition.h
00003  * \brief Highly optimized FPGA vector correlator class
00004  * \authors <ul>
00005  *          <li> Marc Majoral, 2019. mmajoral(at)cttc.cat
00006  *          </ul>
00007  *
00008  * Class that controls and executes a highly optimized acquisition HW
00009  * accelerator in the FPGA
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
```

```
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_FPGA_ACQUISITION_H
00023 #define GNSS_SDR_FPGA_ACQUISITION_H
00024
00025 #include <cstdint>
00026 #include <string>
00027 #include <utility>  // for std::move, std::pair
00028 #include <vector>   // for std::vector
00029
00030 /** \addtogroup Acquisition
00031  * \{ */
00032 /** \addtogroup acquisition_libs
00033  * \{ */
00034
00035
00036 /*!
00037  * \brief Class that implements carrier wipe-off and correlators.
00038  */
00039 class Fpga_Acquisition
00040 {
00041 public:
00042     /*!
00043      * \brief Constructor
00044      */
00045     Fpga_Acquisition(
00046         std::string device_name,
00047         uint32_t select_queue,
00048         std::vector<std::pair<uint32_t, uint32_t» &downsampling_filter_specs,
00049         uint32_t &max_FFT_size);
00050
00051     /*!
00052      * \brief Destructor
00053      */
00054     ~Fpga_Acquisition() = default;
00055
00056     /*!
00057      * \brief Initialize acquisition parameters
00058      */
00059     // void init(uint32_t samples_per_code, uint32_t code_length, int64_t resampled_fs, uint32_t
      *all_fft_codes);
00060     void init(uint32_t nsamples, uint32_t d_fft_size,
00061         int64_t resampled_fs, uint32_t downsampling_filter_num, uint32_t excludelimit, uint32_t
      *all_fft_codes);
00062
00063     /*!
00064      * \brief Select the code with the chosen PRN
00065      */
00066     bool set_local_code(uint32_t PRN);
00067
00068     /*!
00069      * \brief Configure the doppler sweep parameters in the FPGA
00070      */
00071     void set_doppler_sweep(uint32_t num_sweeps, uint32_t doppler_step, int32_t doppler_min);
00072
00073     /*!
00074      * \brief Run the acquisition process in the FPGA
00075      */
00076     void run_acquisition();
00077
00078     /*!
00079      * \brief Read the results of the acquisition process
00080      */
00081     void read_acquisition_results(
00082         uint32_t *max_index,
00083         float *firstpeak,
00084         float *secondpeak,
00085         uint64_t *initial_sample,
00086         float *power_sum,
00087         uint32_t *doppler_index,
00088         uint32_t *total_blk_exp);
00089
00090     /*!
00091      * \brief Reset the FPGA PL.
00092      */
00093     void reset_acquisition();
00094
00095     /*!
00096      * \brief stop the acquisition and the FPGA modules.
00097      */
```

```
00098     void stop_acquisition();
00099
00100     /*!
00101      * \brief Set the block exponent of the FFT in the FPGA.
00102      */
00103     void set_block_exp(uint32_t total_block_exp);
00104
00105     /*!
00106      * \brief Write the PRN code in the FPGA
00107      */
00108     void write_local_code(void);
00109
00110     /*!
00111      * \brief Write the acquisition parameters into the FPGA
00112      */
00113     void configure_acquisition(void);
00114
00115     /*!
00116      * \brief Open the device driver
00117      */
00118     void open_device();
00119
00120     /*!
00121      * \brief Close the device driver
00122      */
00123     void close_device();
00124
00125 private:
00126     // FPGA IP Core version
00127     static const uint32_t FPGA_ACQ_IP_VERSION_1_1 = 0x0001;  // FPGA IP core version 1.1
00128
00129     // FPGA register addresses
00130
00131     // write-only registers
00132     static const uint32_t FREQ_BAND_DOWNSAMPLE_REG_ADDR = 0;      // Select frequency band and
      downsampling filter
00133     static const uint32_t FFT_LENGTH_REG_ADDR = 1;                // Length of the FFT
00134     static const uint32_t CORR_NSAMPLES_REG_ADDR = 2;             // Correlation length
00135     static const uint32_t DOPPLER_MIN_REG_ADDR = 3;               // Doppler min
00136     static const uint32_t DOPPLER_STEP_REG_ADDR = 4;              // Doppler step
00137     static const uint32_t NUM_DOPPLER_SEARCH_STEPS_REG_ADDR = 5;  // Number of Doppler search steps
00138     static const uint32_t PROG_MEM_ADDR = 6;                      // Access to the memory storing the
      PRN code of the target satellite.
00139     static const uint32_t LOG2_FFT_LENGTH_REG_ADDR = 7;           // Log2(FFT_LENGTH)
00140     static const uint32_t ACQ_COMMAND_FLAGS_REG_ADDR = 8;         // Flags that reset, start, and stop
      the acquisition process.
00141     static const uint32_t CLEAR_MEM_REG_ADDR = 9;                 // Flag that resets the write
      address of the PRN code memory.
00142     static const uint32_t MAX_FFT_SCALING_FACTOR_REG_ADDR = 11;   // Reference FFT scaling factor
00143     static const uint32_t EXCL_LIM_REG_ADDR = 12;                 // Exclude Limit value for the
      second FFT peak search process
00144
00145     // read-write registers
00146     static const uint32_t TEST_REG_ADDR = 15;
00147
00148     // read-only registers
00149     static const uint32_t RESULT_VALID_REG_ADDR = 0;                      // Flag that indicates a
      valid result
00150     static const uint32_t SAMPLESTAMP_LSW_REG_ADDR = 1;                   // Sample stamp LSW
00151     static const uint32_t SAMPLESTAMP_MSW_REG_ADDR = 2;                   // Sample stamp MSW
00152     static const uint32_t MAG_SQ_FIRST_PEAK_REG_ADDR = 3;                 // magnitude squared of the
      first peak
00153     static const uint32_t MAG_SQ_SECOND_PEAK_REG_ADDR = 4;                // magnitude squared of the
      second peak
00154     static const uint32_t ACQ_DELAY_SAMPLES_REG_ADDR = 5;                 // acquisition delay in
      samples
00155     static const uint32_t DOPPLER_INDEX_REG_ADDR = 7;                     // Doppler index
00156     static const uint32_t FFT_SCALING_FACTOR_REG_ADDR = 8;                // Scaling factor applied by
      the FFT
00157     static const uint32_t MAX_FFT_SIZE_REG_ADDR = 9;                      // Maximum FFT size
      supported by the FPGA
00158     static const uint32_t DOWNSAMPLING_FILTER_DEC_FACTORS_REG_ADDR = 10;  // Available decimation
      factors
00159     static const uint32_t DOWNSAMPLING_FILTER_LATENCIES_REG_ADDR = 11;    // Available downsampling
      filter latencies
00160     static const uint32_t FPGA_IP_CORE_VERSION_REG_ADDR = 14;             // FPGA acquisition IP core
      version
00161
00162     // FPGA register parameters
00163     static const uint32_t FPGA_PAGE_SIZE = 0x1000;                // default page size for the
      multicorrelator memory map
00164     static const uint32_t LAUNCH_ACQUISITION = 1;                 // command to launch the acquisition
      process
00165     static const uint32_t RESET_ACQUISITION = 2;                  // command to reset the acquisition
      and the FPGA Modules
00166     static const uint32_t STOP_ACQUISITION = 4;                   // command to stop the acquisition
      and the FPGA modules
```

```
00167     static const uint32_t TEST_REG_SANITY_CHECK = 0x55AA;         // value to check the presence of
      the test register (to detect the hw)
00168     static const uint32_t LOCAL_CODE_CLEAR_MEM = 0x10000000;      // command to clear the internal
      memory of the multicorrelator
00169     static const uint32_t MEM_LOCAL_CODE_WR_ENABLE = 0x0C000000;  // command to enable the ENA and WR
      pins of the internal memory of the multicorrelator
00170     static const uint32_t POW_2_2 = 4;                            // 2^2 (used for the conversion of
      floating point numbers to integers)
00171     static const uint32_t POW_2_31 = 2147483648;                  // 2^31 (used for the conversion of
      floating point numbers to integers)
00172     static const uint32_t MAX_FILTERS_AVAILABLE = 2;              // maximum number of downsampling
      filters available in the FPGA by default
00173     static const uint32_t DEFAULT_MAX_FFT_SIZE = 32768;           // default maximum FFT size
      supported by the FPGA
00174     static const uint32_t ACQ_BUFF_0 = 0;                         // FPGA Acquisition IP buffer
      containing L1/E1 frequency band samples by default.
00175     static const uint32_t ACQ_BUFF_1 = 0;                         // FPGA Acquisition IP buffer
      containing L2 or L5/E5a frequency band samples by default.
00176
00177     // bit manipulation
00178     static const uint32_t RSHIFT_4_BITS = 0x4;
00179     static const uint32_t RSHIFT_8_BITS = 0x8;
00180     static const uint32_t BIT_MASK_4 = 0xF;
00181     static const uint32_t BIT_MASK_8 = 0xFF;
00182
00183     // Downsampling default constants
00184     const uint32_t DEFAULT_DOWNSAMPLING_FILTER_DELAY = 40;  // default downsampling filter delay (for
      FPGA Acquisition IP core versions earlier than FPGA_ACQ_IP_VERSION_1)
00185     const uint32_t DEFAULT_DOWNSAMPLING_FACTOR = 4;         // default downsampling factor (for FPGA
      Acquisition IP core versions earlier than FPGA_ACQ_IP_VERSION_1)
00186
00187     // private methods
00188     void fpga_acquisition_test_register(void);
00189     void read_ipcore_info(std::vector<std::pair<uint32_t, uint32_t>> &downsampling_filter_specs,
      uint32_t &max_FFT_size);
00190
00191     std::vector<std::pair<uint32_t, uint32_t>> d_downsampling_filter_specs;
00192     std::string d_device_name;      // HW device name
00193     int64_t d_resampled_fs;         // sampling frequency
00194     volatile uint32_t *d_map_base;  // driver memory map
00195     uint32_t *d_all_fft_codes;      // memory that contains all the code ffts
00196     int32_t d_fd;                   // driver descriptor
00197     uint32_t d_fft_size;            // number of samples including padding
00198     uint32_t d_excludelimit;
00199     uint32_t d_nsamples;                  // number of samples not including padding
00200     uint32_t d_filter_num;                // Selected downsampling filter
00201     uint32_t d_downsampling_factor;       // downsampling_factor
00202     uint32_t d_downsampling_filter_delay;  // Impulse response delay of the downsampling filter
00203     uint32_t d_select_queue;              // queue selection
00204     uint32_t d_PRN;                       // PRN
00205     uint32_t d_IP_core_version;           // FPGA acquisition IP core version
00206 };
00207
00208
00209 /** \} */
00210 /** \} */
00211 #endif  // GNSS_SDR_FPGA_ACQUISITION_H
```

13.36 channel.h File Reference

Interface of a GNSS channel.

```
#include "channel_fsm.h"
#include "channel_interface.h"
#include "channel_msg_receiver_cc.h"
#include "concurrent_queue.h"
#include "gnss_signal.h"
#include "gnss_synchro.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
```

Classes

- class Channel

  This class represents a GNSS channel. It wraps an AcquisitionInterface, a TrackingInterface and a TelemetryDecoderInterface, and handles their interaction through a Finite State Machine.

13.36.1 Detailed Description

Interface of a GNSS channel.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com Luis Esteve, 2011. luis(at)epsilon-formacion.com

It holds blocks for acquisition, tracking, navigation data extraction and pseudorange calculation.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file channel.h.

13.37 channel.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel.h
00003  * \brief Interface of a GNSS channel.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *         Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *
00007  * It holds blocks for acquisition, tracking,
00008  * navigation data extraction and pseudorange calculation.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CHANNEL_H
00022 #define GNSS_SDR_CHANNEL_H
00023
00024 #include "channel_fsm.h"
00025 #include "channel_interface.h"
00026 #include "channel_msg_receiver_cc.h"
00027 #include "concurrent_queue.h"
00028 #include "gnss_signal.h"
00029 #include "gnss_synchro.h"
00030 #include <gnuradio/block.h>
00031 #include <pmt/pmt.h>
00032 #include <cstddef>
00033 #include <cstdint>
00034 #include <memory>
00035 #include <mutex>
00036 #include <string>
00037
00038 /** \addtogroup Channel
00039  * Classes containing a GNSS channel.
00040  * \{ */
00041 /** \addtogroup Channel_adapters channel_adapters
00042  * Classes that wrap an AcquisitionInterface,
00043  * a TrackingInterface and a TelemetryDecoderInterface, and handles
00044  * their interaction.
00045  * \{ */
00046
00047
00048 class ConfigurationInterface;
00049 class AcquisitionInterface;
00050 class TrackingInterface;
00051 class TelemetryDecoderInterface;
00052
00053
00054 /*!
00055  * \brief This class represents a GNSS channel. It wraps an AcquisitionInterface,
00056  * a TrackingInterface and a TelemetryDecoderInterface, and handles
```

```
00057  * their interaction through a Finite State Machine
00058  *
00059  */
00060 class Channel : public ChannelInterface
00061 {
00062 public:
00063     //! Constructor
00064     Channel(const ConfigurationInterface* configuration,
00065         uint32_t channel,
00066         std::shared_ptr<AcquisitionInterface> acq,
00067         std::shared_ptr<TrackingInterface> trk,
00068         std::shared_ptr<TelemetryDecoderInterface> nav,
00069         const std::string& role,
00070         const std::string& signal_str,
00071         Concurrent_Queue<pmt::pmt_t>* queue);
00072
00073     ~Channel() = default;  //!< Destructor
00074
00075     void connect(gr::top_block_sptr top_block) override;  //!< Connects the tracking block to the
      top_block and to the telemetry
00076     void disconnect(gr::top_block_sptr top_block) override;
00077     gr::basic_block_sptr get_left_block() override;
00078     gr::basic_block_sptr get_left_block_trk() override;   //!< Gets the GNU Radio tracking block input
      pointer
00079     gr::basic_block_sptr get_right_block_trk() override;  //!< Gets the GNU Radio tracking block
      output pointer
00080     gr::basic_block_sptr get_left_block_acq() override;   //!< Gets the GNU Radio acquisition block
      input pointer
00081     gr::basic_block_sptr get_right_block_acq() override;  //!< Gets the GNU Radio acquisition block
      output pointer
00082     gr::basic_block_sptr get_right_block() override;      //!< Gets the GNU Radio channel block output
      pointer
00083
00084     inline std::string role() override { return role_; }
00085     inline std::string implementation() override { return std::string("Channel"); }  //!< Returns
      "Channel"
00086     inline size_t item_size() override { return 2 * sizeof(float); }
00087     Gnss_Signal get_signal() override;
00088     void start_acquisition() override;                          //!< Start the State Machine
00089     void stop_channel() override;                               //!< Stop the State Machine
00090     void set_signal(const Gnss_Signal& gnss_signal_) override;  //!< Sets the channel GNSS signal
00091
00092     void assist_acquisition_doppler(double Carrier_Doppler_hz, uint32_t doppler_num_bins = 0)
      override;
00093
00094     inline std::shared_ptr<AcquisitionInterface> acquisition() const { return acq_; }
00095     inline std::shared_ptr<TrackingInterface> tracking() const { return trk_; }
00096     inline std::shared_ptr<TelemetryDecoderInterface> telemetry() const { return nav_; }
00097
00098 private:
00099     std::shared_ptr<ChannelFsm> channel_fsm_;
00100     std::shared_ptr<AcquisitionInterface> acq_;
00101     std::shared_ptr<TrackingInterface> trk_;
00102     std::shared_ptr<TelemetryDecoderInterface> nav_;
00103     channel_msg_receiver_cc_sptr channel_msg_rx_;
00104     Gnss_Synchro gnss_synchro_{};
00105     Gnss_Signal gnss_signal_;
00106     std::string role_;
00107     std::mutex mx_;
00108     uint32_t channel_;
00109     int glonass_extend_correlation_ms_;
00110     bool connected_;
00111     bool repeat_;
00112     bool flag_enable_fpga_;
00113 };
00114
00115
00116 /** \} */
00117 /** \} */
00118 #endif  // GNSS_SDR_CHANNEL_H
```

13.38 channel_fsm.h File Reference

Interface of the State Machine for channel.

```
#include "acquisition_interface.h"
#include "concurrent_queue.h"
#include "telemetry_decoder_interface.h"
#include "tracking_interface.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
```

```
#include <mutex>
```

Classes

- class ChannelFsm

  This class implements a State Machine for channel.

13.38.1 Detailed Description

Interface of the State Machine for channel.

Authors

Javier Arribas, 2019. javiarribas@gmail.com Antonio Ramos, 2017. antonio.ramos(at)cttc.es Luis Esteve, 2011. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file channel_fsm.h.

13.39 channel_fsm.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel_fsm.h
00003  * \brief Interface of the State Machine for channel
00004  * \authors Javier Arribas, 2019. javiarribas@gmail.com
00005  *          Antonio Ramos, 2017. antonio.ramos(at)cttc.es
00006  *          Luis Esteve,   2011. luis(at)epsilon-formacion.com
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_CHANNEL_FSM_H
00020 #define GNSS_SDR_CHANNEL_FSM_H
00021
00022 #include "acquisition_interface.h"
00023 #include "concurrent_queue.h"
00024 #include "telemetry_decoder_interface.h"
00025 #include "tracking_interface.h"
00026 #include <pmt/pmt.h>
00027 #include <cstdint>
00028 #include <memory>
00029 #include <mutex>
00030
00031 /** \addtogroup Channel
00032  * \{ */
00033 /** \addtogroup Channel_libs channel_libs
00034  * Library with utilities for a GNSS Channel.
00035  * \{ */
00036
00037
00038 /*!
00039  * \brief This class implements a State Machine for channel
00040  */
00041 class ChannelFsm
00042 {
00043 public:
00044     ChannelFsm();
00045     virtual ~ChannelFsm() = default;
00046     explicit ChannelFsm(std::shared_ptr<AcquisitionInterface> acquisition);
00047
00048     void set_acquisition(std::shared_ptr<AcquisitionInterface> acquisition);
00049     void set_tracking(std::shared_ptr<TrackingInterface> tracking);
00050     void set_telemetry(std::shared_ptr<TelemetryDecoderInterface> telemetry);
00051     void set_queue(Concurrent_Queue<pmt::pmt_t>* queue);
00052     void set_channel(uint32_t channel);
00053     void start_acquisition();
```

```
00054
00055     // FSM EVENTS
00056     bool Event_start_acquisition();
00057     bool Event_start_acquisition_fpga();
00058     bool Event_stop_channel();
00059     bool Event_failed_tracking_standby();
00060     virtual bool Event_valid_acquisition();
00061     virtual bool Event_failed_acquisition_repeat();
00062     virtual bool Event_failed_acquisition_no_repeat();
00063
00064 private:
00065     enum fsm_state
00066     {
00067         FSM_STATE_IDLE = 0,
00068         FSM_STATE_ACQUISITION,
00069         FSM_STATE_TRACKING,
00070         FSM_STATE_FAILED_NO_REPEAT,
00071         FSM_STATE_TERMINATING
00072     };
00073     void start_tracking();
00074     void stop_acquisition();
00075     void stop_tracking();
00076     void request_satellite();
00077     void notify_stop_tracking();
00078
00079     std::shared_ptr<AcquisitionInterface> acq_;
00080     std::shared_ptr<TrackingInterface> trk_;
00081     std::shared_ptr<TelemetryDecoderInterface> nav_;
00082
00083     std::mutex mx_;
00084
00085     Concurrent_Queue<pmt::pmt_t>* queue_;
00086
00087     uint32_t channel_;
00088     fsm_state state_;
00089 };
00090
00091
00092 /** \} */
00093 /** \} */
00094 #endif  // GNSS_SDR_CHANNEL_FSM_H
```

13.40 channel_msg_receiver_cc.h File Reference

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

```
#include "channel_fsm.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <memory>
```

Classes

- class channel_msg_receiver_cc

  GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

Typedefs

- using channel_msg_receiver_cc_sptr = gnss_shared_ptr<channel_msg_receiver_cc>

Functions

- channel_msg_receiver_cc_sptr **channel_msg_receiver_make_cc** (std::shared_ptr< ChannelFsm > channel_fsm, bool repeat)

13.40.1 Detailed Description

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

Author

Javier Arribas, 2016. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file channel_msg_receiver_cc.h.

13.41 channel_msg_receiver_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel_msg_receiver_cc.h
00003  * \brief GNU Radio block that receives asynchronous channel messages from acquisition and tracking
      blocks
00004  * \author Javier Arribas, 2016. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_CHANNEL_MSG_RECEIVER_CC_H
00018 #define GNSS_SDR_CHANNEL_MSG_RECEIVER_CC_H
00019
00020 #include "channel_fsm.h"
00021 #include <gnuradio/block.h>
00022 #include <pmt/pmt.h>
00023 #include <memory>
00024
00025 /** \addtogroup Channel
00026  * \{ */
00027 /** \addtogroup Channel_libs
00028  * \{ */
00029
00030
00031 class channel_msg_receiver_cc;
00032
00033 using channel_msg_receiver_cc_sptr = gnss_shared_ptr<channel_msg_receiver_cc>;
00034
00035 channel_msg_receiver_cc_sptr channel_msg_receiver_make_cc(std::shared_ptr<ChannelFsm> channel_fsm,
      bool repeat);
00036
00037 /*!
00038  * \brief GNU Radio block that receives asynchronous channel messages from acquisition and tracking
      blocks
00039  */
00040 class channel_msg_receiver_cc : public gr::block
00041 {
00042 public:
00043     ~channel_msg_receiver_cc() = default;  //!< Default destructor
00044
00045 private:
00046     friend channel_msg_receiver_cc_sptr channel_msg_receiver_make_cc(std::shared_ptr<ChannelFsm>
      channel_fsm, bool repeat);
00047     channel_msg_receiver_cc(std::shared_ptr<ChannelFsm> channel_fsm, bool repeat);
00048     void msg_handler_channel_events(const pmt::pmt_t& msg);
00049     std::shared_ptr<ChannelFsm> d_channel_fsm;
00050     bool d_repeat;  // todo: change FSM to include repeat value
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_CHANNEL_MSG_RECEIVER_CC_H
```

13.42 array_signal_conditioner.h File Reference

It wraps blocks to change data type, filter and resample input data, adapted to array receiver.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <cstddef>
```

```
#include <memory>
#include <string>
```

Classes

- class ArraySignalConditioner

  This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.

13.42.1 Detailed Description

It wraps blocks to change data type, filter and resample input data, adapted to array receiver.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file array_signal_conditioner.h.

13.43 array_signal_conditioner.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file array_signal_conditioner.h
00003  * \brief It wraps blocks to change data type, filter and resample input data, adapted to array
      receiver
00004  * \author Javier Arribas jarribas (at) cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_ARRAY_SIGNAL_CONDITIONER_H
00019 #define GNSS_SDR_ARRAY_SIGNAL_CONDITIONER_H
00020
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/block.h>
00024 #include <cstddef>
00025 #include <memory>
00026 #include <string>
00027
00028 /** \addtogroup Signal_Conditioner
00029  * \{ */
00030 /** \addtogroup Signal_Conditioner_adapters
00031  * \{ */
00032
00033
00034 class ConfigurationInterface;
00035
00036
00037 /*!
00038  * \brief This class wraps blocks to change data_type_adapter, input_filter and resampler
00039  * to be applied to the input flow of sampled signal.
00040  */
00041 class ArraySignalConditioner : public GNSSBlockInterface
00042 {
00043 public:
00044     //! Constructor
00045     ArraySignalConditioner(std::shared_ptr<GNSSBlockInterface> data_type_adapt,
00046         std::shared_ptr<GNSSBlockInterface> in_filt,
00047         std::shared_ptr<GNSSBlockInterface> res,
00048         std::string role);
00049
00050     //! Destructor
```

```
00051     ~ArraySignalConditioner() = default;
00052
00053     void connect(gr::top_block_sptr top_block) override;
00054     void disconnect(gr::top_block_sptr top_block) override;
00055     gr::basic_block_sptr get_left_block() override;
00056     gr::basic_block_sptr get_right_block() override;
00057
00058     inline std::string role() override { return role_; }
00059     //! Returns "Array_Signal_Conditioner"
00060     inline std::string implementation() override { return "Array_Signal_Conditioner"; }
00061     inline size_t item_size() override { return data_type_adapt_->item_size(); }
00062
00063     inline std::shared_ptr<GNSSBlockInterface> data_type_adapter() { return data_type_adapt_; }
00064     inline std::shared_ptr<GNSSBlockInterface> input_filter() { return in_filt_; }
00065     inline std::shared_ptr<GNSSBlockInterface> resampler() { return res_; }
00066
00067 private:
00068     std::shared_ptr<GNSSBlockInterface> data_type_adapt_;
00069     std::shared_ptr<GNSSBlockInterface> in_filt_;
00070     std::shared_ptr<GNSSBlockInterface> res_;
00071     std::string role_;
00072     bool connected_;
00073 };
00074
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_SIGNAL_CONDITIONER_H
```

13.44 signal_conditioner.h File Reference

It wraps blocks to change data type, filter and resample input data.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <cstddef>
#include <memory>
#include <string>
```

Classes

- class SignalConditioner

  This class wraps blocks to change data_type_adapter, input_filter and resampler to be applied to the input flow of sampled signal.

13.44.1 Detailed Description

It wraps blocks to change data type, filter and resample input data.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file signal_conditioner.h.

13.45 signal_conditioner.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_conditioner.h
00003  * \brief It wraps blocks to change data type, filter and resample input data.
00004  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
```

```
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * ------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SIGNAL_CONDITIONER_H
00019 #define GNSS_SDR_SIGNAL_CONDITIONER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/block.h>
00023 #include <cstddef>
00024 #include <memory>
00025 #include <string>
00026
00027 /** \addtogroup Signal_Conditioner Signal Conditioner
00028  * Signal Conditioner wrapper block
00029  * \{ */
00030 /** \addtogroup Signal_Conditioner_adapters conditioner_adapters
00031  * Wrap a Signal Conditioner with a GNSSBlockInterface
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief This class wraps blocks to change data_type_adapter, input_filter and resampler
00037  * to be applied to the input flow of sampled signal.
00038  */
00039 class SignalConditioner : public GNSSBlockInterface
00040 {
00041 public:
00042     //! Constructor
00043     SignalConditioner(std::shared_ptr<GNSSBlockInterface> data_type_adapt,
00044         std::shared_ptr<GNSSBlockInterface> in_filt,
00045         std::shared_ptr<GNSSBlockInterface> res,
00046         std::string role);
00047
00048     //! Destructor
00049     ~SignalConditioner() = default;
00050
00051     void connect(gr::top_block_sptr top_block) override;
00052     void disconnect(gr::top_block_sptr top_block) override;
00053     gr::basic_block_sptr get_left_block() override;
00054     gr::basic_block_sptr get_right_block() override;
00055
00056     inline std::string role() override { return role_; }
00057
00058     inline std::string implementation() override { return "Signal_Conditioner"; }  //!< Returns
      "Signal_Conditioner"
00059
00060     inline size_t item_size() override { return data_type_adapt_->item_size(); }
00061
00062     inline std::shared_ptr<GNSSBlockInterface> data_type_adapter() { return data_type_adapt_; }
00063     inline std::shared_ptr<GNSSBlockInterface> input_filter() { return in_filt_; }
00064     inline std::shared_ptr<GNSSBlockInterface> resampler() { return res_; }
00065
00066 private:
00067     std::shared_ptr<GNSSBlockInterface> data_type_adapt_;
00068     std::shared_ptr<GNSSBlockInterface> in_filt_;
00069     std::shared_ptr<GNSSBlockInterface> res_;
00070     std::string role_;
00071     bool connected_;
00072 };
00073
00074
00075 /** \} */
00076 /** \} */
00077 #endif  // GNSS_SDR_SIGNAL_CONDITIONER_H
```

13.46 byte_to_short.h File Reference

Adapts an 8-bits sample stream (IF) to a short int stream (IF).

```
#include "gnss_block_interface.h"
#include <gnuradio/blocks/char_to_short.h>
#include <gnuradio/blocks/file_sink.h>
#include <cstdint>
#include <string>
```

Classes

- class ByteToShort

  Adapts an 8-bits sample stream (IF) to a short int stream (IF).

13.46.1 Detailed Description

Adapts an 8-bits sample stream (IF) to a short int stream (IF).

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file byte_to_short.h.

13.47 byte_to_short.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file byte_to_short.h
00003  * \brief Adapts an 8-bits sample stream (IF) to a short int stream (IF)
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BYTE_TO_SHORT_H
00018 #define GNSS_SDR_BYTE_TO_SHORT_H
00019
00020 #include "gnss_block_interface.h"
00021 #include <gnuradio/blocks/char_to_short.h>
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <cstdint>
00024 #include <string>
00025
00026 /** \addtogroup Data_Type Data Type Adapters
00027  * Classes for data type conversion
00028  * \{ */
00029 /** \addtogroup Data_type_adapters data_type_adapters
00030  * Wrap GNU Radio data type adapter blocks with a GNSSBlockInterface
00031  * \{ */
00032
00033
00034 class ConfigurationInterface;
00035
00036 /*!
00037  * \brief Adapts an 8-bits sample stream (IF) to a short int stream (IF)
00038  *
00039  */
00040 class ByteToShort : public GNSSBlockInterface
00041 {
00042 public:
00043     ByteToShort(const ConfigurationInterface* configuration,
00044         std::string role, unsigned int in_streams,
00045         unsigned int out_streams);
00046
00047     ~ByteToShort() = default;
00048
00049     inline std::string role() override
00050     {
00051         return role_;
00052     }
00053
00054     //! Returns "Byte_To_Short"
00055     inline std::string implementation() override
00056     {
00057         return "Byte_To_Short";
00058     }
00059
```

```
00060     inline size_t item_size() override
00061     {
00062         return sizeof(int8_t);
00063     }
00064
00065     void connect(gr::top_block_sptr top_block) override;
00066     void disconnect(gr::top_block_sptr top_block) override;
00067     gr::basic_block_sptr get_left_block() override;
00068     gr::basic_block_sptr get_right_block() override;
00069
00070 private:
00071     gr::blocks::char_to_short::sptr gr_char_to_short_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string input_item_type_;
00075     std::string output_item_type_;
00076     std::string role_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool dump_;
00080 };
00081
00082
00083 /** \} */
00084 /** \} */
00085 #endif  // GNSS_SDR_BYTE_TO_SHORT_H
```

13.48 cshort_to_grcomplex.h File Reference

Adapts an 16-bits complex sample stream to a float complex stream.

```
#include "cshort_to_gr_complex.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/file_sink.h>
#include <cstdint>
#include <string>
```

Classes

- class CshortToGrComplex

  Adapts an 16-bits complex sample stream to a float complex stream.

13.48.1 Detailed Description

Adapts an 16-bits complex sample stream to a float complex stream.

Author

Carles Fernandez Prades, 2014 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file cshort_to_grcomplex.h.

13.49 cshort_to_grcomplex.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cshort_to_grcomplex.h
00003  * \brief Adapts an 16-bits complex sample stream to a float complex stream
00004  * \author Carles Fernandez Prades, 2014 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
```

```
00015  */
00016
00017 #ifndef GNSS_SDR_CSHORT_TO_GRCOMPLEX_H
00018 #define GNSS_SDR_CSHORT_TO_GRCOMPLEX_H
00019
00020 #include "cshort_to_gr_complex.h"
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <cstdint>
00024 #include <string>
00025
00026 /** \addtogroup Data_Type Data Type Adapters
00027  * Classes for data type conversion
00028  * \{ */
00029 /** \addtogroup Data_type_adapters data_type_adapters
00030  * Wrap GNU Radio data type adapter blocks with a GNSSBlockInterface
00031  * \{ */
00032
00033
00034 class ConfigurationInterface;
00035
00036 /*!
00037  * \brief Adapts an 16-bits complex sample stream to a float complex stream
00038  *
00039  */
00040 class CshortToGrComplex : public GNSSBlockInterface
00041 {
00042 public:
00043     CshortToGrComplex(const ConfigurationInterface* configuration,
00044         std::string role, unsigned int in_streams,
00045         unsigned int out_streams);
00046
00047     ~CshortToGrComplex() = default;
00048
00049     inline std::string role() override
00050     {
00051         return role_;
00052     }
00053
00054     //! Returns "Cshort_To_Gr_Complex"
00055     inline std::string implementation() override
00056     {
00057         return "Cshort_To_Gr_Complex";
00058     }
00059
00060     inline size_t item_size() override
00061     {
00062         return 2 * sizeof(float);
00063     }
00064
00065     void connect(gr::top_block_sptr top_block) override;
00066     void disconnect(gr::top_block_sptr top_block) override;
00067     gr::basic_block_sptr get_left_block() override;
00068     gr::basic_block_sptr get_right_block() override;
00069
00070 private:
00071     cshort_to_gr_complex_sptr cshort_to_gr_complex_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string role_;
00075     unsigned int in_streams_;
00076     unsigned int out_streams_;
00077     bool dump_;
00078 };
00079
00080
00081 /** \} */
00082 /** \} */
00083 #endif  // GNSS_SDR_CSHORT_TO_GRCOMPLEX_H
```

13.50 ibyte_to_cbyte.h File Reference

Adapts an I/Q interleaved byte (unsigned char) sample stream into a std::complex<unsigned char> stream.

```
#include "conjugate_ic.h"
#include "gnss_block_interface.h"
#include "interleaved_byte_to_complex_byte.h"
#include <gnuradio/blocks/file_sink.h>
#include <cstdint>
#include <string>
```

Classes

- class IbyteToCbyte

13.50.1 Detailed Description

Adapts an I/Q interleaved byte (unsigned char) sample stream into a std::complex<unsigned char> stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ibyte_to_cbyte.h.

13.51 ibyte_to_cbyte.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ibyte_to_cbyte.h
00003  * \brief \brief Adapts an I/Q interleaved byte (unsigned char) sample stream
00004  * into a std::complex<unsigned char> stream
00005  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_IBYTE_TO_CBYTE_H
00019 #define GNSS_SDR_IBYTE_TO_CBYTE_H
00020
00021 #include "conjugate_ic.h"
00022 #include "gnss_block_interface.h"
00023 #include "interleaved_byte_to_complex_byte.h"
00024 #include <gnuradio/blocks/file_sink.h>
00025 #include <cstdint>
00026 #include <string>
00027
00028 /** \addtogroup Data_Type
00029  * \{ */
00030 /** \addtogroup Data_type_adapters
00031  * \{ */
00032
00033
00034 class ConfigurationInterface;
00035
00036 /*!
00037  * \briefAdapts an I/Q interleaved byte (unsigned char) sample stream
00038  * into a std::complex<unsigned char> stream
00039  */
00040 class IbyteToCbyte : public GNSSBlockInterface
00041 {
00042 public:
00043     IbyteToCbyte(const ConfigurationInterface* configuration,
00044         const std::string& role, unsigned int in_streams,
00045         unsigned int out_streams);
00046
00047     ~IbyteToCbyte() = default;
00048
00049     inline std::string role() override
00050     {
00051         return role_;
00052     }
00053
00054     //! Returns "Ibyte_To_Cbyte"
00055     inline std::string implementation() override
00056     {
00057         return "Ibyte_To_Cbyte";
00058     }
00059
00060     inline size_t item_size() override
00061     {
```

```
00062         return 2 * sizeof(int8_t);
00063     }
00064
00065     void connect(gr::top_block_sptr top_block) override;
00066     void disconnect(gr::top_block_sptr top_block) override;
00067     gr::basic_block_sptr get_left_block() override;
00068     gr::basic_block_sptr get_right_block() override;
00069
00070 private:
00071     interleaved_byte_to_complex_byte_sptr ibyte_to_cbyte_;
00072     conjugate_ic_sptr conjugate_ic_;
00073     gr::blocks::file_sink::sptr file_sink_;
00074     std::string dump_filename_;
00075     std::string input_item_type_;
00076     std::string output_item_type_;
00077     std::string role_;
00078     unsigned int in_streams_;
00079     unsigned int out_streams_;
00080     bool inverted_spectrum;
00081     bool dump_;
00082 };
00083
00084
00085 /** \} */
00086 /** \} */
00087 #endif  // GNSS_SDR_IBYTE_TO_CBYTE_H
```

13.52 ibyte_to_complex.h File Reference

Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.

```
#include "conjugate_cc.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/interleaved_char_to_complex.h>
#include <cstdint>
#include <string>
```

Classes

- class IbyteToComplex

  Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.

13.52.1 Detailed Description

Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ibyte_to_complex.h.

13.53 ibyte_to_complex.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ibyte_to_complex.h
00003  * \brief Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
```

```
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_IBYTE_TO_COMPLEX_H
00018 #define GNSS_SDR_IBYTE_TO_COMPLEX_H
00019
00020 #include "conjugate_cc.h"
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <gnuradio/blocks/interleaved_char_to_complex.h>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Data_Type
00028  * \{ */
00029 /** \addtogroup Data_type_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 /*!
00036  * \brief Adapts an I/Q interleaved byte integer sample stream to a gr_complex (float) stream
00037  *
00038  */
00039 class IbyteToComplex : public GNSSBlockInterface
00040 {
00041 public:
00042     IbyteToComplex(const ConfigurationInterface* configuration,
00043         const std::string& role, unsigned int in_streams,
00044         unsigned int out_streams);
00045
00046     ~IbyteToComplex() = default;
00047
00048     inline std::string role() override
00049     {
00050         return role_;
00051     }
00052
00053     //! Returns "Ibyte_To_Complex"
00054     inline std::string implementation() override
00055     {
00056         return "Ibyte_To_Complex";
00057     }
00058
00059     inline size_t item_size() override
00060     {
00061         return 2 * sizeof(int8_t);
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069 private:
00070     gr::blocks::interleaved_char_to_complex::sptr gr_interleaved_char_to_complex_;
00071     conjugate_cc_sptr conjugate_cc_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string input_item_type_;
00075     std::string output_item_type_;
00076     std::string role_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool inverted_spectrum;
00080     bool dump_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_IBYTE_TO_COMPLEX_H
```

13.54 ibyte_to_cshort.h File Reference

Adapts a short interleaved sample stream into a std::complex<short> stream.

```
#include "conjugate_sc.h"
#include "gnss_block_interface.h"
#include "interleaved_byte_to_complex_short.h"
#include <gnuradio/blocks/file_sink.h>
#include <cstdint>
```

```
#include <string>
```

Classes

- class IbyteToCshort

  Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.

13.54.1 Detailed Description

Adapts a short interleaved sample stream into a std::complex<short> stream.

Author

Carles Fernandez-Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ibyte_to_cshort.h.

13.55 ibyte_to_cshort.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ibyte_to_cshort.h
00003  * \brief Adapts a short interleaved sample stream into a std::complex<short> stream
00004  * \author Carles Fernandez-Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_IBYTE_TO_CSHORT_H
00018 #define GNSS_SDR_IBYTE_TO_CSHORT_H
00019
00020 #include "conjugate_sc.h"
00021 #include "gnss_block_interface.h"
00022 #include "interleaved_byte_to_complex_short.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Data_Type
00028  * \{ */
00029 /** \addtogroup Data_type_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 /*!
00036  * \brief Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream
00037  *
00038  */
00039 class IbyteToCshort : public GNSSBlockInterface
00040 {
00041 public:
00042     IbyteToCshort(const ConfigurationInterface* configuration,
00043         const std::string& role, unsigned int in_streams,
00044         unsigned int out_streams);
00045
00046     ~IbyteToCshort() = default;
00047
00048     inline std::string role() override
00049     {
00050         return role_;
00051     }
00052
00053     //! Returns "Ibyte_To_Cshort"
00054     inline std::string implementation() override
```

```
00055     {
00056         return "Ibyte_To_Cshort";
00057     }
00058
00059     inline size_t item_size() override
00060     {
00061         return 2 * sizeof(int8_t);
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069 private:
00070     interleaved_byte_to_complex_short_sptr interleaved_byte_to_complex_short_;
00071     conjugate_sc_sptr conjugate_sc_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string input_item_type_;
00075     std::string output_item_type_;
00076     std::string role_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool inverted_spectrum;
00080     bool dump_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_IBYTE_TO_CSHORT_H
```

13.56 ishort_to_complex.h File Reference

Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.

```
#include "conjugate_cc.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/interleaved_short_to_complex.h>
#include <cstdint>
#include <string>
```

Classes

- class IshortToComplex

  Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.

13.56.1 Detailed Description

Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ishort_to_complex.h.

13.57 ishort_to_complex.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ishort_to_complex.h
00003  * \brief Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
```

```
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_ISHORT_TO_COMPLEX_H
00018 #define GNSS_SDR_ISHORT_TO_COMPLEX_H
00019
00020 #include "conjugate_cc.h"
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <gnuradio/blocks/interleaved_short_to_complex.h>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Data_Type
00028  * \{ */
00029 /** \addtogroup Data_type_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 /*!
00036  * \brief Adapts an I/Q interleaved short integer sample stream to a gr_complex (float) stream
00037  *
00038  */
00039 class IshortToComplex : public GNSSBlockInterface
00040 {
00041 public:
00042     IshortToComplex(const ConfigurationInterface* configuration,
00043         const std::string& role, unsigned int in_streams,
00044         unsigned int out_streams);
00045
00046     ~IshortToComplex() = default;
00047
00048     inline std::string role() override
00049     {
00050         return role_;
00051     }
00052
00053     //! Returns "Ishort_To_Complex"
00054     inline std::string implementation() override
00055     {
00056         return "Ishort_To_Complex";
00057     }
00058
00059     inline size_t item_size() override
00060     {
00061         return 2 * sizeof(int16_t);
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069 private:
00070     gr::blocks::interleaved_short_to_complex::sptr gr_interleaved_short_to_complex_;
00071     conjugate_cc_sptr conjugate_cc_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string input_item_type_;
00075     std::string output_item_type_;
00076     std::string role_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool inverted_spectrum;
00080     bool dump_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_ISHORT_TO_COMPLEX_H
```

13.58 ishort_to_cshort.h File Reference

Adapts a short interleaved sample stream into a std::complex<short> stream.

```
#include "conjugate_sc.h"
#include "gnss_block_interface.h"
#include "interleaved_short_to_complex_short.h"
#include <gnuradio/blocks/file_sink.h>
#include <cstdint>
#include <string>
```

Classes

- class IshortToCshort

  Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream.

13.58.1 Detailed Description

Adapts a short interleaved sample stream into a std::complex<short> stream.

Author

Carles Fernandez-Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ishort_to_cshort.h.

13.59 ishort_to_cshort.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ishort_to_cshort.h
00003  * \brief Adapts a short interleaved sample stream into a std::complex<short> stream
00004  * \author Carles Fernandez-Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_ISHORT_TO_CSHORT_H
00018 #define GNSS_SDR_ISHORT_TO_CSHORT_H
00019
00020 #include "conjugate_sc.h"
00021 #include "gnss_block_interface.h"
00022 #include "interleaved_short_to_complex_short.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Data_Type
00028  * \{ */
00029 /** \addtogroup Data_type_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 /*!
00036  * \brief Adapts a short integer (16 bits) interleaved sample stream into a std::complex<short> stream
00037  *
00038  */
00039 class IshortToCshort : public GNSSBlockInterface
00040 {
00041 public:
00042     IshortToCshort(const ConfigurationInterface* configuration,
00043         const std::string& role, unsigned int in_streams,
00044         unsigned int out_streams);
00045
00046     ~IshortToCshort() = default;
```

```
00047
00048     inline std::string role() override
00049     {
00050         return role_;
00051     }
00052
00053     //! Returns "Ishort_To_Cshort"
00054     inline std::string implementation() override
00055     {
00056         return "Ishort_To_Cshort";
00057     }
00058
00059     inline size_t item_size() override
00060     {
00061         return 2 * sizeof(int16_t);
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069 private:
00070     interleaved_short_to_complex_short_sptr interleaved_short_to_complex_short_;
00071     conjugate_sc_sptr conjugate_sc_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string input_item_type_;
00075     std::string output_item_type_;
00076     std::string role_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool inverted_spectrum;
00080     bool dump_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_ISHORT_TO_CSHORT_H
```

13.60 cshort_to_gr_complex.h File Reference

Adapts a complex short (16 + 16 bits) sample stream into a std::complex<float> stream (32 + 32 bits).

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
```

Classes

- class cshort_to_gr_complex

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<float> stream.

Typedefs

- using cshort_to_gr_complex_sptr = gnss_shared_ptr<cshort_to_gr_complex>

Functions

- cshort_to_gr_complex_sptr **make_cshort_to_gr_complex** ()

13.60.1 Detailed Description

Adapts a complex short (16 + 16 bits) sample stream into a std::complex<float> stream (32 + 32 bits).

Author

Carles Fernandez Prades, 2014 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cshort_to_gr_complex.h.

13.61 cshort_to_gr_complex.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cshort_to_gr_complex.h
00003  * \brief Adapts a complex short (16 + 16 bits) sample stream into a
00004  *   std::complex<float> stream (32 + 32 bits)
00005  * \author Carles Fernandez Prades, 2014 cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CSHORT_TO_GR_COMPLEX_H
00019 #define GNSS_SDR_CSHORT_TO_GR_COMPLEX_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023
00024 /** \addtogroup Data_Type
00025  * \{ */
00026 /** \addtogroup data_type_gnuradio_blocks
00027  * \{ */
00028
00029
00030 class cshort_to_gr_complex;
00031
00032 using cshort_to_gr_complex_sptr = gnss_shared_ptr<cshort_to_gr_complex>;
00033
00034 cshort_to_gr_complex_sptr make_cshort_to_gr_complex();
00035
00036 /*!
00037  * \brief This class adapts a short (16-bits) interleaved sample stream
00038  * into a std::complex<float> stream
00039  */
00040 class cshort_to_gr_complex : public gr::sync_block
00041 {
00042 public:
00043     int work(int noutput_items,
00044         gr_vector_const_void_star &input_items,
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     friend cshort_to_gr_complex_sptr make_cshort_to_gr_complex();
00049     cshort_to_gr_complex();
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_CSHORT_TO_GR_COMPLEX_H
```

13.62 interleaved_byte_to_complex_byte.h File Reference

Adapts an 8-bits interleaved sample stream into a 16-bits complex stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_decimator.h>
```

Classes

- class interleaved_byte_to_complex_byte

  This class adapts an 8-bits interleaved sample stream into a 16-bits complex stream (std::complex<unsigned char>).

Typedefs

- using interleaved_byte_to_complex_byte_sptr = gnss_shared_ptr<interleaved_byte_to_complex_byte>

Functions

- interleaved_byte_to_complex_byte_sptr **make_interleaved_byte_to_complex_byte** ()

13.62.1 Detailed Description

Adapts an 8-bits interleaved sample stream into a 16-bits complex stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file interleaved_byte_to_complex_byte.h.

13.63 interleaved_byte_to_complex_byte.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file interleaved_byte_to_complex_byte.h
00003  * \brief Adapts an 8-bits interleaved sample stream into a 16-bits complex stream
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_BYTE_H
00018 #define GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_BYTE_H
00019
00020 #include "gnss_block_interface.h"
00021 #include <gnuradio/sync_decimator.h>
00022
00023 /** \addtogroup Data_Type
00024  * \{ */
00025 /** \addtogroup data_type_gnuradio_blocks data_type_gr_blocks
00026  * GNU Radio Blocks for data type conversion
00027  * \{ */
00028
00029
00030 class interleaved_byte_to_complex_byte;
00031
00032 using interleaved_byte_to_complex_byte_sptr = gnss_shared_ptr<interleaved_byte_to_complex_byte>;
00033
00034 interleaved_byte_to_complex_byte_sptr make_interleaved_byte_to_complex_byte();
00035
00036 /*!
00037  * \brief This class adapts an 8-bits interleaved sample stream
00038  * into a 16-bits complex stream (std::complex<unsigned char>)
00039  */
00040 class interleaved_byte_to_complex_byte : public gr::sync_decimator
00041 {
00042 public:
00043     int work(int noutput_items,
00044         gr_vector_const_void_star &input_items,
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     friend interleaved_byte_to_complex_byte_sptr make_interleaved_byte_to_complex_byte();
00049     interleaved_byte_to_complex_byte();
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_BYTE_H
```

13.64 interleaved_byte_to_complex_short.h File Reference

Adapts a byte (8-bits) interleaved sample stream into a std::complex<short> stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_decimator.h>
```

Classes

- class interleaved_byte_to_complex_short

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.

Typedefs

- using interleaved_byte_to_complex_short_sptr = gnss_shared_ptr<interleaved_byte_to_complex_short>

Functions

- interleaved_byte_to_complex_short_sptr **make_interleaved_byte_to_complex_short** ()

13.64.1 Detailed Description

Adapts a byte (8-bits) interleaved sample stream into a std::complex<short> stream.

Author

Javier Arribas (jarribas(at)cttc.es)

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file interleaved_byte_to_complex_short.h.

13.65 interleaved_byte_to_complex_short.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file interleaved_byte_to_complex_short.h
00003  * \brief Adapts a byte (8-bits) interleaved sample stream into a std::complex<short> stream
00004  * \author Javier Arribas (jarribas(at)cttc.es)
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_SHORT_H
00018 #define GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_SHORT_H
00019
00020 #include "gnss_block_interface.h"
00021 #include <gnuradio/sync_decimator.h>
00022
00023
00024 /** \addtogroup Data_Type
00025  * \{ */
00026 /** \addtogroup data_type_gnuradio_blocks
00027  * \{ */
00028
00029
00030 class interleaved_byte_to_complex_short;
00031
00032 using interleaved_byte_to_complex_short_sptr = gnss_shared_ptr<interleaved_byte_to_complex_short>;
00033
00034 interleaved_byte_to_complex_short_sptr make_interleaved_byte_to_complex_short();
00035
00036 /*!
00037  * \brief This class adapts a short (16-bits) interleaved sample stream
00038  * into a std::complex<short> stream
00039  */
00040 class interleaved_byte_to_complex_short : public gr::sync_decimator
00041 {
```

```
00042 public:
00043     int work(int noutput_items,
00044         gr_vector_const_void_star &input_items,
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     friend interleaved_byte_to_complex_short_sptr make_interleaved_byte_to_complex_short();
00049     interleaved_byte_to_complex_short();
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_INTERLEAVED_BYTE_TO_COMPLEX_SHORT_H
```

13.66 interleaved_short_to_complex_short.h File Reference

Adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_decimator.h>
```

Classes

- class interleaved_short_to_complex_short

  This class adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.

Typedefs

- using interleaved_short_to_complex_short_sptr = gnss_shared_ptr<interleaved_short_to_complex_short>

Functions

- interleaved_short_to_complex_short_sptr **make_interleaved_short_to_complex_short** ()

13.66.1 Detailed Description

Adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file interleaved_short_to_complex_short.h.

13.67 interleaved_short_to_complex_short.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file interleaved_short_to_complex_short.h
00003  * \brief Adapts a short (16-bits) interleaved sample stream into a std::complex<short> stream
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_INTERLEAVED_SHORT_TO_COMPLEX_SHORT_H
00018 #define GNSS_SDR_INTERLEAVED_SHORT_TO_COMPLEX_SHORT_H
00019
```

```
00020 #include "gnss_block_interface.h"
00021 #include <gnuradio/sync_decimator.h>
00022
00023 /** \addtogroup Data_Type
00024  * \{ */
00025 /** \addtogroup data_type_gnuradio_blocks
00026  * \{ */
00027
00028
00029 class interleaved_short_to_complex_short;
00030
00031 using interleaved_short_to_complex_short_sptr = gnss_shared_ptr<interleaved_short_to_complex_short>;
00032
00033 interleaved_short_to_complex_short_sptr make_interleaved_short_to_complex_short();
00034
00035 /*!
00036  * \brief This class adapts a short (16-bits) interleaved sample stream
00037  * into a std::complex<short> stream
00038  */
00039 class interleaved_short_to_complex_short : public gr::sync_decimator
00040 {
00041 public:
00042     int work(int noutput_items,
00043         gr_vector_const_void_star &input_items,
00044         gr_vector_void_star &output_items);
00045
00046 private:
00047     friend interleaved_short_to_complex_short_sptr make_interleaved_short_to_complex_short();
00048     interleaved_short_to_complex_short();
00049 };
00050
00051
00052 /** \} */
00053 /** \} */
00054 #endif  // GNSS_SDR_INTERLEAVED_SHORT_TO_COMPLEX_SHORT_H
```

13.68 beamformer_filter.h File Reference

Interface of an adapter of a digital beamformer.

```
#include "gnss_block_interface.h"
#include <gnuradio/hier_block2.h>
#include <cstdint>
#include <string>
```

Classes

- class BeamformerFilter

  Interface of an adapter of a digital beamformer block to a GNSSBlockInterface.

13.68.1 Detailed Description

Interface of an adapter of a digital beamformer.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beamformer_filter.h.

13.69 beamformer_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beamformer_filter.h
00003  * \brief Interface of an adapter of a digital beamformer
00004  * \author Javier Arribas jarribas (at) cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
```

```
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_BEAMFORMER_FILTER_H
00019 #define GNSS_SDR_BEAMFORMER_FILTER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/hier_block2.h>
00023 #include <cstdint>
00024 #include <string>
00025
00026 /** \addtogroup Input_Filter
00027  * \{ */
00028 /** \addtogroup Input_filter_adapters
00029  * \{ */
00030
00031
00032 class ConfigurationInterface;
00033
00034 /*!
00035  * \brief Interface of an adapter of a digital beamformer block
00036  * to a GNSSBlockInterface
00037  */
00038 class BeamformerFilter : public GNSSBlockInterface
00039 {
00040 public:
00041     BeamformerFilter(const ConfigurationInterface* configuration,
00042         const std::string& role, unsigned int in_stream,
00043         unsigned int out_stream);
00044
00045     ~BeamformerFilter() = default;
00046
00047     inline std::string role() override
00048     {
00049         return role_;
00050     }
00051
00052     //! returns "Beamformer_Filte"
00053     inline std::string implementation() override
00054     {
00055         return "Beamformer_Filter";
00056     }
00057
00058     inline size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block() override;
00067
00068 private:
00069     gr::block_sptr beamformer_;
00070     gr::block_sptr file_sink_;
00071     std::string role_;
00072     std::string item_type_;
00073     std::string dump_filename_;
00074     size_t item_size_;
00075     unsigned int in_stream_;
00076     unsigned int out_stream_;
00077     bool dump_;
00078 };
00079
00080
00081 /** \} */
00082 /** \} */
00083 #endif  // GNSS_SDR_BEAMFORMER_FILTER_H
```

13.70 fir_filter.h File Reference

Adapts a gnuradio gr_fir_filter designed with pm_remez.

```
#include "byte_x2_to_complex_byte.h"
#include "complex_byte_to_float_x2.h"
#include "cshort_to_float_x2.h"
```

```
#include "gnss_block_interface.h"
#include "short_x2_to_cshort.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/float_to_char.h>
#include <gnuradio/blocks/float_to_complex.h>
#include <gnuradio/blocks/float_to_short.h>
#include <gnuradio/gr_complex.h>
#include <gnuradio/filter/fir_filter_ccf.h>
#include <gnuradio/filter/fir_filter_fff.h>
#include <cmath>
#include <string>
#include <vector>
```

Classes

- class FirFilter

  This class adapts a GNU Radio gr_fir_filter designed with pm_remez.

13.70.1 Detailed Description

Adapts a gnuradio gr_fir_filter designed with pm_remez.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fir_filter.h.

13.71 fir_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fir_filter.h
00003  * \brief Adapts a gnuradio gr_fir_filter designed with pm_remez
00004  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_FIR_FILTER_H
00019 #define GNSS_SDR_FIR_FILTER_H
00020
00021 #include "byte_x2_to_complex_byte.h"
00022 #include "complex_byte_to_float_x2.h"
00023 #include "cshort_to_float_x2.h"
00024 #include "gnss_block_interface.h"
00025 #include "short_x2_to_cshort.h"
00026 #include <gnuradio/blocks/file_sink.h>
00027 #include <gnuradio/blocks/float_to_char.h>
00028 #include <gnuradio/blocks/float_to_complex.h>
00029 #include <gnuradio/blocks/float_to_short.h>
00030 #include <gnuradio/gr_complex.h>
00031 #ifdef GR_GREATER_38
00032 #include <gnuradio/filter/fir_filter_blk.h>
00033 #else
00034 #include <gnuradio/filter/fir_filter_ccf.h>
00035 #include <gnuradio/filter/fir_filter_fff.h>
00036 #endif
00037 #include <cmath>
```

```
00038 #include <string>
00039 #include <vector>
00040
00041 /** \addtogroup Input_Filter Input Filter
00042  * Classes for input signal filtering
00043  * \{ */
00044 /** \addtogroup Input_filter_adapters input_filter_adapters
00045  * Classes that wrap GNU Radio input filters with a GNSSBlockInterface
00046  * \{ */
00047
00048
00049 class ConfigurationInterface;
00050
00051 /*!
00052  * \brief This class adapts a GNU Radio gr_fir_filter designed with pm_remez
00053  *
00054  * See Parks-McClellan FIR filter design,
      https://en.wikipedia.org/wiki/Parks-McClellan_filter_design_algorithm
00055  * Calculates the optimal (in the Chebyshev/minimax sense) FIR filter impulse response
00056  * given a set of band edges, the desired response on those bands, and the weight given
00057  * to the error in those bands.
00058  */
00059 class FirFilter : public GNSSBlockInterface
00060 {
00061 public:
00062     //! Constructor
00063     FirFilter(const ConfigurationInterface* configuration,
00064         std::string role,
00065         unsigned int in_streams,
00066         unsigned int out_streams);
00067
00068     //! Destructor
00069     ~FirFilter() = default;
00070
00071     inline std::string role() override
00072     {
00073         return role_;
00074     }
00075
00076     //! Returns "Fir_Filter"
00077     inline std::string implementation() override
00078     {
00079         return "Fir_Filter";
00080     }
00081
00082     inline size_t item_size() override
00083     {
00084         return item_size_;
00085     }
00086
00087     void connect(gr::top_block_sptr top_block) override;
00088     void disconnect(gr::top_block_sptr top_block) override;
00089     gr::basic_block_sptr get_left_block() override;
00090     gr::basic_block_sptr get_right_block() override;
00091
00092 private:
00093     void init();
00094
00095     gr::filter::fir_filter_ccf::sptr fir_filter_ccf_;
00096     gr::filter::fir_filter_fff::sptr fir_filter_fff_1_;
00097     gr::filter::fir_filter_fff::sptr fir_filter_fff_2_;
00098     gr::blocks::float_to_complex::sptr float_to_complex_;
00099     gr::blocks::float_to_short::sptr float_to_short_1_;
00100     gr::blocks::float_to_short::sptr float_to_short_2_;
00101     short_x2_to_cshort_sptr short_x2_to_cshort_;
00102     complex_byte_to_float_x2_sptr cbyte_to_float_x2_;
00103     byte_x2_to_complex_byte_sptr char_x2_cbyte_;
00104     cshort_to_float_x2_sptr cshort_to_float_x2_;
00105     gr::blocks::float_to_char::sptr float_to_char_1_;
00106     gr::blocks::float_to_char::sptr float_to_char_2_;
00107     gr::blocks::file_sink::sptr file_sink_;
00108     const ConfigurationInterface* config_;
00109     std::vector<float> taps_;
00110     std::string dump_filename_;
00111     std::string input_item_type_;
00112     std::string output_item_type_;
00113     std::string taps_item_type_;
00114     std::string role_;
00115     size_t item_size_;
00116     unsigned int in_streams_;
00117     unsigned int out_streams_;
00118     bool dump_;
00119 };
00120
00121
00122 /** \} */
00123 /** \} */
```

```
00124 #endif  // GNSS_SDR_FIR_FILTER_H
```

13.72 freq_xlating_fir_filter.h File Reference

Adapts a gnuradio gr_freq_xlating_fir_filter designed with gr_remez.

```
#include "complex_float_to_complex_byte.h"
#include "gnss_block_interface.h"
#include "short_x2_to_cshort.h"
#include <gnuradio/filter/freq_xlating_fir_filter_ccf.h>
#include <gnuradio/filter/freq_xlating_fir_filter_fcf.h>
#include <gnuradio/filter/freq_xlating_fir_filter_scf.h>
#include <gnuradio/blocks/char_to_short.h>
#include <gnuradio/blocks/complex_to_float.h>
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/float_to_short.h>
#include <string>
#include <vector>
```

Classes

- class FreqXlatingFirFilter

  This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez.

13.72.1 Detailed Description

Adapts a gnuradio gr_freq_xlating_fir_filter designed with gr_remez.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file freq_xlating_fir_filter.h.

13.73 freq_xlating_fir_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file freq_xlating_fir_filter.h
00003  * \brief Adapts a gnuradio gr_freq_xlating_fir_filter designed with gr_remez
00004  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_FREQ_XLATING_FIR_FILTER_H
00019 #define GNSS_SDR_FREQ_XLATING_FIR_FILTER_H
00020
00021 #include "complex_float_to_complex_byte.h"
00022 #include "gnss_block_interface.h"
00023 #include "short_x2_to_cshort.h"
00024 #ifdef GR_GREATER_38
00025 #include <gnuradio/filter/freq_xlating_fir_filter.h>
00026 #else
00027 #include <gnuradio/filter/freq_xlating_fir_filter_ccf.h>
00028 #include <gnuradio/filter/freq_xlating_fir_filter_fcf.h>
00029 #include <gnuradio/filter/freq_xlating_fir_filter_scf.h>
```

```
00030 #endif
00031 #include <gnuradio/blocks/char_to_short.h>
00032 #include <gnuradio/blocks/complex_to_float.h>
00033 #include <gnuradio/blocks/file_sink.h>
00034 #include <gnuradio/blocks/float_to_short.h>
00035 #include <string>
00036 #include <vector>
00037
00038 /** \addtogroup Input_Filter
00039  * \{ */
00040 /** \addtogroup Input_filter_adapters
00041  * \{ */
00042
00043
00044 class ConfigurationInterface;
00045
00046 /*!
00047  * \brief This class adapts a gnuradio gr_freq_xlating_fir_filter designed with pm_remez
00048  *
00049  * Construct a FIR filter with the given taps and a composite frequency
00050  * translation that shifts intermediate_freq_ down to zero Hz. The frequency
00051  * translation logically comes before the filtering operation.
00052  *
00053  * See Parks-McClellan FIR filter design,
      https://en.wikipedia.org/wiki/Parks-McClellan_filter_design_algorithm
00054  * Calculates the optimal (in the Chebyshev/minimax sense) FIR filter impulse response
00055  * given a set of band edges, the desired response on those bands, and the weight given
00056  * to the error in those bands.
00057  */
00058 class FreqXlatingFirFilter : public GNSSBlockInterface
00059 {
00060 public:
00061     FreqXlatingFirFilter(const ConfigurationInterface* configuration,
00062         std::string role, unsigned int in_streams,
00063         unsigned int out_streams);
00064
00065     ~FreqXlatingFirFilter() = default;
00066
00067     inline std::string role() override
00068     {
00069         return role_;
00070     }
00071
00072     //! Returns "Freq_Xlating_Fir_Filter"
00073     inline std::string implementation() override
00074     {
00075         return "Freq_Xlating_Fir_Filter";
00076     }
00077
00078     inline size_t item_size() override
00079     {
00080         return input_size_;
00081     }
00082
00083     void connect(gr::top_block_sptr top_block) override;
00084     void disconnect(gr::top_block_sptr top_block) override;
00085     gr::basic_block_sptr get_left_block() override;
00086     gr::basic_block_sptr get_right_block() override;
00087
00088 private:
00089     gr::filter::freq_xlating_fir_filter_ccf::sptr freq_xlating_fir_filter_ccf_;
00090     gr::filter::freq_xlating_fir_filter_fcf::sptr freq_xlating_fir_filter_fcf_;
00091     gr::filter::freq_xlating_fir_filter_scf::sptr freq_xlating_fir_filter_scf_;
00092     gr::blocks::complex_to_float::sptr complex_to_float_;
00093     gr::blocks::char_to_short::sptr gr_char_to_short_;
00094     gr::blocks::float_to_short::sptr float_to_short_1_;
00095     gr::blocks::float_to_short::sptr float_to_short_2_;
00096     short_x2_to_cshort_sptr short_x2_to_cshort_;
00097     complex_float_to_complex_byte_sptr complex_to_complex_byte_;
00098     gr::blocks::file_sink::sptr file_sink_;
00099     std::vector<float> taps_;
00100     std::string dump_filename_;
00101     std::string input_item_type_;
00102     std::string output_item_type_;
00103     std::string taps_item_type_;
00104     std::string role_;
00105     size_t input_size_;
00106     double intermediate_freq_;
00107     double sampling_freq_;
00108     int decimation_factor_;
00109     unsigned int in_streams_;
00110     unsigned int out_streams_;
00111     bool dump_;
00112 };
00113
00114
00115 /** \} */
```

```
00116 /** \} */
00117 #endif  // GNSS_SDR_FREQ_XLATING_FIR_FILTER_H
```

13.74 notch_filter.h File Reference

Adapter of a multistate Notch filter.

```
#include "gnss_block_interface.h"
#include "notch_cc.h"
#include <gnuradio/blocks/file_sink.h>
#include <string>
#include <vector>
```

Classes

- class NotchFilter

13.74.1 Detailed Description

Adapter of a multistate Notch filter.

Author

Antonio Ramos, 2017. antonio.ramosdet(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file notch_filter.h.

13.75 notch_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file notch_filter.h
00003  * \brief Adapter of a multistate Notch filter
00004  * \author Antonio Ramos, 2017. antonio.ramosdet(at)gmail.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_NOTCH_FILTER_H
00019 #define GNSS_SDR_NOTCH_FILTER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "notch_cc.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <string>
00025 #include <vector>
00026
00027 /** \addtogroup Input_Filter
00028  * \{ */
00029 /** \addtogroup Input_filter_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 class NotchFilter : public GNSSBlockInterface
00036 {
00037 public:
00038     NotchFilter(const ConfigurationInterface* configuration,
00039         const std::string& role, unsigned int in_streams,
00040         unsigned int out_streams);
```

```
00041
00042     ~NotchFilter() = default;
00043
00044     std::string role()
00045     {
00046         return role_;
00047     }
00048
00049     //! Returns "Notch_Filter"
00050     std::string implementation()
00051     {
00052         return "Notch_Filter";
00053     }
00054
00055     size_t item_size()
00056     {
00057         return item_size_;
00058     }
00059
00060     void connect(gr::top_block_sptr top_block);
00061     void disconnect(gr::top_block_sptr top_block);
00062     gr::basic_block_sptr get_left_block();
00063     gr::basic_block_sptr get_right_block();
00064
00065 private:
00066     notch_sptr notch_filter_;
00067     gr::blocks::file_sink::sptr file_sink_;
00068     std::string dump_filename_;
00069     std::string role_;
00070     std::string item_type_;
00071     size_t item_size_;
00072     unsigned int in_streams_;
00073     unsigned int out_streams_;
00074     bool dump_;
00075 };
00076
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_NOTCH_FILTER_H
```

13.76 notch_filter_lite.h File Reference

Adapts a light version of a multistate notch filter.

```
#include "gnss_block_interface.h"
#include "notch_lite_cc.h"
#include <gnuradio/blocks/file_sink.h>
#include <string>
#include <vector>
```

Classes

- class NotchFilterLite

13.76.1 Detailed Description

Adapts a light version of a multistate notch filter.

Author

Antonio Ramos, 2017. antonio.ramosdet(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file notch_filter_lite.h.

13.77 notch_filter_lite.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file notch_filter_lite.h
```

```
00003  * \brief Adapts a light version of a multistate notch filter
00004  * \author Antonio Ramos, 2017. antonio.ramosdet(at)gmail.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_NOTCH_FILTER_LITE_H
00019 #define GNSS_SDR_NOTCH_FILTER_LITE_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "notch_lite_cc.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <string>
00025 #include <vector>
00026
00027 /** \addtogroup Input_Filter
00028  * \{ */
00029 /** \addtogroup Input_filter_adapters
00030  * \{ */
00031
00032
00033 class ConfigurationInterface;
00034
00035 class NotchFilterLite : public GNSSBlockInterface
00036 {
00037 public:
00038     NotchFilterLite(const ConfigurationInterface* configuration,
00039         const std::string& role, unsigned int in_streams,
00040         unsigned int out_streams);
00041
00042     ~NotchFilterLite() = default;
00043
00044     std::string role()
00045     {
00046         return role_;
00047     }
00048
00049     //! Returns "Notch_Filter_Lite"
00050     std::string implementation()
00051     {
00052         return "Notch_Filter_Lite";
00053     }
00054
00055     size_t item_size()
00056     {
00057         return item_size_;
00058     }
00059
00060     void connect(gr::top_block_sptr top_block);
00061     void disconnect(gr::top_block_sptr top_block);
00062     gr::basic_block_sptr get_left_block();
00063     gr::basic_block_sptr get_right_block();
00064
00065 private:
00066     notch_lite_sptr notch_filter_lite_;
00067     gr::blocks::file_sink::sptr file_sink_;
00068     std::string dump_filename_;
00069     std::string role_;
00070     std::string item_type_;
00071     size_t item_size_;
00072     unsigned int in_streams_;
00073     unsigned int out_streams_;
00074     bool dump_;
00075 };
00076
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_NOTCH_FILTER_LITE_H
```

13.78 pulse_blanking_filter.h File Reference

Instantiates the GNSS-SDR pulse blanking filter.

```
#include "gnss_block_interface.h"
#include "pulse_blanking_cc.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/filter/freq_xlating_fir_filter_ccf.h>
#include <string>
```

Classes

- class PulseBlankingFilter

13.78.1 Detailed Description

Instantiates the GNSS-SDR pulse blanking filter.

Author

Javier Arribas 2017 Antonio Ramos 2017

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pulse_blanking_filter.h.

13.79 pulse_blanking_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pulse_blanking_filter.h
00003  * \brief Instantiates the GNSS-SDR pulse blanking filter
00004  * \author Javier Arribas 2017
00005  *         Antonio Ramos  2017
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_PULSE_BLANKING_FILTER_H
00019 #define GNSS_SDR_PULSE_BLANKING_FILTER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "pulse_blanking_cc.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #ifdef GR_GREATER_38
00025 #include <gnuradio/filter/freq_xlating_fir_filter.h>
00026 #else
00027 #include <gnuradio/filter/freq_xlating_fir_filter_ccf.h>
00028 #endif
00029 #include <string>
00030
00031 /** \addtogroup Input_Filter
00032  * \{ */
00033 /** \addtogroup Input_filter_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 class PulseBlankingFilter : public GNSSBlockInterface
00040 {
00041 public:
00042     PulseBlankingFilter(const ConfigurationInterface* configuration,
00043         std::string role, unsigned int in_streams,
00044         unsigned int out_streams);
00045
00046     ~PulseBlankingFilter() = default;
00047
00048     inline std::string role() override
00049     {
00050         return role_;
```

```
00051     }
00052
00053     //! Returns "Pulse_Blanking_Filter"
00054     inline std::string implementation() override
00055     {
00056         return "Pulse_Blanking_Filter";
00057     }
00058
00059     inline size_t item_size() override
00060     {
00061         return input_size_;
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069 private:
00070     pulse_blanking_cc_sptr pulse_blanking_cc_;
00071     gr::filter::freq_xlating_fir_filter_ccf::sptr freq_xlating_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073     std::string dump_filename_;
00074     std::string item_type_;
00075     std::string role_;
00076     size_t input_size_;
00077     unsigned int in_streams_;
00078     unsigned int out_streams_;
00079     bool dump_;
00080     bool xlat_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_PULSE_BLANKING_FILTER_H
```

13.80 beamformer.h File Reference

Simple spatial filter using RAW array input and beamforming coefficients.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <vector>
```

Classes

- class beamformer

  This class implements a real-time software-defined spatial filter using the CTTC GNSS experimental antenna array input and a set of dynamically reloadable weights.

Typedefs

- using beamformer_sptr = gnss_shared_ptr<beamformer>

Functions

- beamformer_sptr **make_beamformer_sptr** ()

Variables

- const int GNSS_SDR_BEAMFORMER_CHANNELS = 8

13.80.1 Detailed Description

Simple spatial filter using RAW array input and beamforming coefficients.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beamformer.h.

13.81 beamformer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beamformer.h
00003  *
00004  * \brief Simple spatial filter using RAW array input and beamforming coefficients
00005  * \author Javier Arribas jarribas (at) cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_BEAMFORMER_H
00019 #define GNSS_SDR_BEAMFORMER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <vector>
00024
00025 /** \addtogroup Input_Filter
00026  * \{ */
00027 /** \addtogroup Input_filter_gnuradio_blocks
00028  * \{ */
00029
00030
00031 class beamformer;
00032
00033 using beamformer_sptr = gnss_shared_ptr<beamformer>;
00034
00035 beamformer_sptr make_beamformer_sptr();
00036
00037 const int GNSS_SDR_BEAMFORMER_CHANNELS = 8;
00038
00039 /*!
00040  * \brief This class implements a real-time software-defined spatial filter using the CTTC GNSS
      experimental antenna array input and a set of dynamically reloadable weights
00041  */
00042 class beamformer : public gr::sync_block
00043 {
00044 public:
00045     ~beamformer() = default;
00046     int work(int noutput_items, gr_vector_const_void_star &input_items,
00047         gr_vector_void_star &output_items);
00048
00049 private:
00050     friend beamformer_sptr make_beamformer_sptr();
00051     beamformer();
00052     std::vector<gr_complex> weight_vector = std::vector<gr_complex>(GNSS_SDR_BEAMFORMER_CHANNELS,
      gr_complex(1.0, 0.0));
00053 };
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_BEAMFORMER_H
```

13.82 notch_cc.h File Reference

Implements a notch filter algorithm.

```
#include "gnss_block_interface.h"
#include "gnss_sdr_fft.h"
#include <gnuradio/block.h>
```

```
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstdint>
#include <memory>
```

Classes

- class Notch

  This class implements a real-time software-defined multi state notch filter.

Typedefs

- using notch_sptr = gnss_shared_ptr<Notch>

Functions

- notch_sptr **make_notch_filter** (float pfa, float p_c_factor, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)

13.82.1 Detailed Description

Implements a notch filter algorithm.

Author

Antonio Ramos (antonio.ramosdet(at)gmail.com)

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file notch_cc.h.

13.83 notch_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file notch_cc.h
00003  * \brief Implements a notch filter algorithm
00004  * \author Antonio Ramos (antonio.ramosdet(at)gmail.com)
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  */
00017
00018 #ifndef GNSS_SDR_NOTCH_CC_H
00019 #define GNSS_SDR_NOTCH_CC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "gnss_sdr_fft.h"
00023 #include <gnuradio/block.h>
00024 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00025 #include <cstdint>
00026 #include <memory>
00027
00028 /** \addtogroup Input_Filter
00029  * \{ */
00030 /** \addtogroup Input_filter_gnuradio_blocks
00031  * \{ */
00032
00033
00034 class Notch;
00035
00036 using notch_sptr = gnss_shared_ptr<Notch>;
00037
00038 notch_sptr make_notch_filter(
```

```
00039     float pfa,
00040     float p_c_factor,
00041     int32_t length,
00042     int32_t n_segments_est,
00043     int32_t n_segments_reset);
00044
00045 /*!
00046  * \brief This class implements a real-time software-defined multi state notch filter
00047  */
00048 class Notch : public gr::block
00049 {
00050 public:
00051     ~Notch() = default;
00052
00053     int general_work(int noutput_items, gr_vector_int &ninput_items,
00054         gr_vector_const_void_star &input_items,
00055         gr_vector_void_star &output_items);
00056
00057 private:
00058     friend notch_sptr make_notch_filter(float pfa, float p_c_factor, int32_t length, int32_t
      n_segments_est, int32_t n_segments_reset);
00059     Notch(float pfa, float p_c_factor, int32_t length, int32_t n_segments_est, int32_t
      n_segments_reset);
00060
00061     std::unique_ptr<gnss_fft_complex_fwd> d_fft_;
00062     volk_gnsssdr::vector<gr_complex> c_samples_;
00063     volk_gnsssdr::vector<float> angle_;
00064     volk_gnsssdr::vector<float> power_spect_;
00065     gr_complex last_out_;
00066     gr_complex z_0_;
00067     gr_complex p_c_factor_;
00068     float pfa_;
00069     float noise_pow_est_;
00070     float thres_;
00071     int32_t length_;
00072     int32_t n_deg_fred_;
00073     uint32_t n_segments_;
00074     uint32_t n_segments_est_;
00075     uint32_t n_segments_reset_;
00076     bool filter_state_;
00077 };
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_NOTCH_CC_H
```

13.84 notch_lite_cc.h File Reference

Implements a notch filter light algorithm.

```
#include "gnss_block_interface.h"
#include "gnss_sdr_fft.h"
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstdint>
#include <memory>
```

Classes

- class NotchLite

  This class implements a real-time software-defined multi state notch filter light version.

Typedefs

- using notch_lite_sptr = gnss_shared_ptr<NotchLite>

Functions

- notch_lite_sptr **make_notch_filter_lite** (float p_c_factor, float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset, int32_t n_segments_coeff)

13.84.1 Detailed Description

Implements a notch filter light algorithm.

Author

Antonio Ramos (antonio.ramosdet(at)gmail.com)

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file notch_lite_cc.h.

13.85 notch_lite_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file notch_lite_cc.h
00003  * \brief Implements a notch filter light algorithm
00004  * \author Antonio Ramos (antonio.ramosdet(at)gmail.com)
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  */
00017
00018 #ifndef GNSS_SDR_NOTCH_LITE_CC_H
00019 #define GNSS_SDR_NOTCH_LITE_CC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "gnss_sdr_fft.h"
00023 #include <gnuradio/block.h>
00024 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00025 #include <cstdint>
00026 #include <memory>
00027
00028 /** \addtogroup Input_Filter
00029  * \{ */
00030 /** \addtogroup Input_filter_gnuradio_blocks
00031  * \{ */
00032
00033
00034 class NotchLite;
00035
00036 using notch_lite_sptr = gnss_shared_ptr<NotchLite>;
00037
00038 notch_lite_sptr make_notch_filter_lite(
00039     float p_c_factor,
00040     float pfa,
00041     int32_t length,
00042     int32_t n_segments_est,
00043     int32_t n_segments_reset,
00044     int32_t n_segments_coeff);
00045
00046 /*!
00047  * \brief This class implements a real-time software-defined multi state notch filter light version
00048  */
00049 class NotchLite : public gr::block
00050 {
00051 public:
00052     ~NotchLite() = default;
00053
00054     int general_work(int noutput_items, gr_vector_int &ninput_items,
00055         gr_vector_const_void_star &input_items,
00056         gr_vector_void_star &output_items);
00057
00058 private:
00059     friend notch_lite_sptr make_notch_filter_lite(float p_c_factor, float pfa, int32_t length, int32_t
      n_segments_est, int32_t n_segments_reset, int32_t n_segments_coeff);
00060     NotchLite(float p_c_factor, float pfa, int32_t length, int32_t n_segments_est, int32_t
      n_segments_reset, int32_t n_segments_coeff);
00061
00062     std::unique_ptr<gnss_fft_complex_fwd> d_fft_;
00063     volk_gnsssdr::vector<float> power_spect_;
00064     gr_complex last_out_;
00065     gr_complex z_0_;
```

```
00066     gr_complex p_c_factor_;
00067     gr_complex c_samples1_;
00068     gr_complex c_samples2_;
00069     float pfa_;
00070     float thres_;
00071     float noise_pow_est_;
00072     float angle1_;
00073     float angle2_;
00074     int32_t length_;
00075     int32_t n_segments_;
00076     int32_t n_segments_est_;
00077     int32_t n_segments_reset_;
00078     int32_t n_segments_coeff_reset_;
00079     int32_t n_segments_coeff_;
00080     int32_t n_deg_fred_;
00081     bool filter_state_;
00082 };
00083
00084
00085 /** \} */
00086 /** \} */
00087 #endif  // GNSS_SDR_NOTCH_LITE_CC_H
```

13.86 pulse_blanking_cc.h File Reference

Implements a pulse blanking algorithm.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstdint>
```

Classes

- class pulse_blanking_cc

Typedefs

- using pulse_blanking_cc_sptr = gnss_shared_ptr<pulse_blanking_cc>

Functions

- pulse_blanking_cc_sptr **make_pulse_blanking_cc** (float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset)

13.86.1 Detailed Description

Implements a pulse blanking algorithm.

Author

Javier Arribas (jarribas(at)cttc.es) Antonio Ramos (antonio.ramosdet(at)gmail.com)

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pulse_blanking_cc.h.

13.87 pulse_blanking_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pulse_blanking_cc.h
00003  * \brief Implements a pulse blanking algorithm
00004  * \author Javier Arribas (jarribas(at)cttc.es)
00005  *         Antonio Ramos  (antonio.ramosdet(at)gmail.com)
00006  *
00007  * -----------------------------------------------------------------------
00008  *
```

```
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -------------------------------------------------------------------------
00016  *
00017  */
00018
00019 #ifndef GNSS_SDR_PULSE_BLANKING_CC_H
00020 #define GNSS_SDR_PULSE_BLANKING_CC_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/block.h>
00024 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00025 #include <cstdint>
00026
00027 /** \addtogroup Input_Filter
00028  * \{ */
00029 /** \addtogroup Input_filter_gnuradio_blocks input_filter_gr_blocks
00030  * GNU Radio blocks implementing input filters,
00031  * \{ */
00032
00033
00034 class pulse_blanking_cc;
00035
00036 using pulse_blanking_cc_sptr = gnss_shared_ptr<pulse_blanking_cc>;
00037
00038 pulse_blanking_cc_sptr make_pulse_blanking_cc(
00039     float pfa,
00040     int32_t length,
00041     int32_t n_segments_est,
00042     int32_t n_segments_reset);
00043
00044 class pulse_blanking_cc : public gr::block
00045 {
00046 public:
00047     ~pulse_blanking_cc() = default;
00048
00049     int general_work(int noutput_items __attribute__((unused)), gr_vector_int &ninput_items
      __attribute__((unused)),
00050         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00051
00052 private:
00053     friend pulse_blanking_cc_sptr make_pulse_blanking_cc(float pfa, int32_t length, int32_t
      n_segments_est, int32_t n_segments_reset);
00054     pulse_blanking_cc(float pfa, int32_t length, int32_t n_segments_est, int32_t n_segments_reset);
00055     volk_gnsssdr::vector<gr_complex> zeros_;
00056     float noise_power_estimation_;
00057     float thres_;
00058     float pfa_;
00059     int32_t length_;
00060     int32_t n_segments_;
00061     int32_t n_segments_est_;
00062     int32_t n_segments_reset_;
00063     int32_t n_deg_fred_;
00064     bool last_filtered_;
00065 };
00066
00067
00068 /** \} */
00069 /** \} */
00070 #endif  // GNSS_SDR_PULSE_BLANKING_CC_H
```

13.88 beidou_b1c_signal_replica.h File Reference

Library for BeiDou B1C signal replica generation.

```
#include <array>
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void **beidou_b1c_code_gen_int** (own::span< int32_t > dest, const std::array< char, 3 > &signal_id, int32↩ _t prn)

- void **beidou_b1c_code_gen_sinboc11_float** (own::span< float > dest, const std::array< char, 3 > &signal_id, uint32_t prn)
- void **beidou_b1c_code_gen_float_sampled** (own::span< float > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)
- void **beidou_b1c_code_gen_complex_sampled** (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)
- void **beidou_b1c_code_gen_complex_sampled** (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

13.88.1 Detailed Description

Library for BeiDou B1C signal replica generation.

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_b1c_signal_replica.h.

13.89 beidou_b1c_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b1c_signal_replica.h
00003  * \brief Library for BeiDou B1C signal replica generation
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_B1C_SIGNAL_REPLICA_H
00017 #define GNSS_SDR_BEIDOU_B1C_SIGNAL_REPLICA_H
00018
00019 #include <array>
00020 #include <complex>
00021 #include <cstdint>
00022 #if HAS_STD_SPAN
00023 #include <span>
00024 namespace own = std;
00025 #else
00026 #include <gsl-lite/gsl-lite.hpp>
00027 namespace own = gsl_lite;
00028 #endif
00029
00030 /** \addtogroup Algorithms_Library
00031  * \{ */
00032 /** \addtogroup Algorithm_libs algorithms_libs
00033  * \{ */
00034
00035 void beidou_b1c_code_gen_int(own::span<int32_t> dest, const std::array<char, 3>& signal_id, int32_t
      prn);
00036 void beidou_b1c_code_gen_sinboc11_float(own::span<float> dest, const std::array<char, 3>& signal_id,
      uint32_t prn);
00037 void beidou_b1c_code_gen_float_sampled(own::span<float> dest, const std::array<char, 3>& signal_id,
00038     bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag);
00039 void beidou_b1c_code_gen_complex_sampled(own::span<std::complex<float» dest, const std::array<char,
      3>& signal_id,
00040     bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag);
00041 void beidou_b1c_code_gen_complex_sampled(own::span<std::complex<float» dest, const std::array<char,
      3>& signal_id,
00042     bool qmboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift);
00043
00044 /** \} */
00045 /** \} */
00046 #endif  // GNSS_SDR_BEIDOU_B1C_SIGNAL_REPLICA_H
```

13.90 beidou_b1i_signal_replica.h File Reference

This file implements various functions for BeiDou B1I signal replica generation.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void beidou_b1i_code_gen_int (own::span< int32_t > dest, int32_t prn, uint32_t chip_shift)

  Generates int32_t GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip↩ _shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift.
- void beidou_b1i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.

13.90.1 Detailed Description

This file implements various functions for BeiDou B1I signal replica generation.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_b1i_signal_replica.h.

13.91 beidou_b1i_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b1i_signal_replica.h
00003  * \brief This file implements various functions for BeiDou B1I signal replica
00004  * generation
00005  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_BEIDOU_B1I_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_BEIDOU_B1I_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #if HAS_STD_SPAN
00025 #include <span>
00026 namespace own = std;
00027 #else
00028 #include <gsl-lite/gsl-lite.hpp>
00029 namespace own = gsl_lite;
00030 #endif
00031
00032 /** \addtogroup Algorithms_Library Algorithms Common Library
```

```
00033  * Common utilities for the GNSS receiver.
00034  * \{ */
00035 /** \addtogroup Algorithm_libs algorithms_libs
00036  * Common utilities for GNSS algorithms.
00037  * \{ */
00038
00039
00040 //! Generates int32_t GPS L1 C/A code for the desired SV ID and code shift
00041 void beidou_b1i_code_gen_int(own::span<int32_t> dest, int32_t prn, uint32_t chip_shift);
00042
00043 //! Generates float GPS L1 C/A code for the desired SV ID and code shift
00044 void beidou_b1i_code_gen_float(own::span<float> dest, int32_t prn, uint32_t chip_shift);
00045
00046 //! Generates complex GPS L1 C/A code for the desired SV ID and code shift
00047 void beidou_b1i_code_gen_complex(own::span<std::complex<float» dest, int32_t prn, uint32_t
      chip_shift);
00048
00049 //! Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific
      sampling frequency
00050 void beidou_b1i_code_gen_complex_sampled(own::span<std::complex<float» dest, uint32_t prn, int32_t
      sampling_freq, uint32_t chip_shift);
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_BEIDOU_B1I_SIGNAL_REPLICA_H
```

13.92 beidou_b2a_signal_replica.h File Reference

BeiDou B2a ranging-code replica (BDS-SIS-ICD-B2a-1.0 Tables 5-2, 5-3).

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void **beidou_b2a_data_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_pilot_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_int** (own::span< int > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_float** (own::span< float > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_complex** (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip_shift)
- void **beidou_b2a_code_gen_complex_sampled** (own::span< std::complex< float > > dest, uint32_t prn, int sampling_freq, uint32_t chip_shift)

13.92.1 Detailed Description

BeiDou B2a ranging-code replica (BDS-SIS-ICD-B2a-1.0 Tables 5-2, 5-3).

Author

huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_b2a_signal_replica.h.

13.93 beidou_b2a_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b2a_signal_replica.h
00003  * \brief BeiDou B2a ranging-code replica (BDS-SIS-ICD-B2a-1.0 Tables 5-2, 5-3)
00004  * \author huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_B2A_SIGNAL_REPLICA_H
00018 #define GNSS_SDR_BEIDOU_B2A_SIGNAL_REPLICA_H
00019
00020 #include <complex>
00021 #include <cstdint>
00022 #if HAS_STD_SPAN
00023 #include <span>
00024 namespace own = std;
00025 #else
00026 #include <gsl-lite/gsl-lite.hpp>
00027 namespace own = gsl_lite;
00028 #endif
00029
00030 /** \addtogroup Algorithms_Library
00031  * \{ */
00032 /** \addtogroup Algorithm_libs algorithms_libs
00033  * \{ */
00034
00035 void beidou_b2a_data_code_gen_int(own::span<int> dest, int32_t prn, uint32_t chip_shift);
00036 void beidou_b2a_pilot_code_gen_int(own::span<int> dest, int32_t prn, uint32_t chip_shift);
00037 void beidou_b2a_code_gen_int(own::span<int> dest, int32_t prn, uint32_t chip_shift);
00038 void beidou_b2a_code_gen_float(own::span<float> dest, int32_t prn, uint32_t chip_shift);
00039 void beidou_b2a_code_gen_complex(own::span<std::complex<float» dest, int32_t prn, uint32_t
      chip_shift);
00040 void beidou_b2a_code_gen_complex_sampled(own::span<std::complex<float» dest, uint32_t prn, int
      sampling_freq, uint32_t chip_shift);
00041
00042 /** \} */
00043 /** \} */
00044 #endif  // GNSS_SDR_BEIDOU_B2A_SIGNAL_REPLICA_H
```

13.94 beidou_b3i_signal_replica.h File Reference

This file implements various functions for BeiDou B3I signal replica generation.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void beidou_b3i_code_gen_int (own::span< int > dest, int32_t prn, uint32_t chip_shift)

  Generates int BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip↩ _shift)

  Generates complex BeiDou B3I code for the desired SV ID and code shift.
- void beidou_b3i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int sampling_freq, uint32_t chip_shift)

  Generates complex BeiDou B3I code for the desired SV ID and code shift, and sampled to specific sampling frequency.

13.94.1 Detailed Description

This file implements various functions for BeiDou B3I signal replica generation.

Author

Damian Miralles, 2019. dmiralles2009@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_b3i_signal_replica.h.

13.95 beidou_b3i_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b3i_signal_replica.h
00003  * \brief This file implements various functions for BeiDou B3I signal replica
00004  * generation
00005  * \author Damian Miralles, 2019. dmiralles2009@gmail.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_BEIDOU_B3I_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_BEIDOU_B3I_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #if HAS_STD_SPAN
00025 #include <span>
00026 namespace own = std;
00027 #else
00028 #include <gsl-lite/gsl-lite.hpp>
00029 namespace own = gsl_lite;
00030 #endif
00031
00032 /** \addtogroup Algorithms_Library
00033  * \{ */
00034 /** \addtogroup Algorithm_libs algorithms_libs
00035  * \{ */
00036
00037
00038 //! Generates int BeiDou B3I code for the desired SV ID and code shift
00039 void beidou_b3i_code_gen_int(own::span<int> dest, int32_t prn, uint32_t chip_shift);
00040
00041 //! Generates float BeiDou B3I code for the desired SV ID and code shift
00042 void beidou_b3i_code_gen_float(own::span<float> dest, int32_t prn, uint32_t chip_shift);
00043
00044 //! Generates complex BeiDou B3I code for the desired SV ID and code shift
00045 void beidou_b3i_code_gen_complex(own::span<std::complex<float>» dest, int32_t prn, uint32_t
      chip_shift);
00046
00047 //! Generates complex BeiDou B3I code for the desired SV ID and code shift, and sampled to specific
      sampling frequency
00048 void beidou_b3i_code_gen_complex_sampled(own::span<std::complex<float>» dest, uint32_t prn, int
      sampling_freq, uint32_t chip_shift);
00049
00050
00051 /** \} */
00052 /** \} */
00053 #endif  // GNSS_SDR_BEIDOU_B3I_SIGNAL_REPLICA_H
```

13.96 byte_x2_to_complex_byte.h File Reference

Adapts two signed char streams into a std::complex<signed char> stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class byte_x2_to_complex_byte

  This class adapts two signed char streams into a std::complex<signed char> stream.

Typedefs

- using byte_x2_to_complex_byte_sptr = gnss_shared_ptr<byte_x2_to_complex_byte>

Functions

- byte_x2_to_complex_byte_sptr **make_byte_x2_to_complex_byte** ()

13.96.1 Detailed Description

Adapts two signed char streams into a std::complex<signed char> stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file byte_x2_to_complex_byte.h.

13.97 byte_x2_to_complex_byte.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file byte_x2_to_complex_byte.h
00003  * \brief Adapts two signed char streams into a std::complex<signed char> stream
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_BYTE_X2_TO_COMPLEX_BYTE_H
00019 #define GNSS_SDR_BYTE_X2_TO_COMPLEX_BYTE_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class byte_x2_to_complex_byte;
00032
00033 using byte_x2_to_complex_byte_sptr = gnss_shared_ptr<byte_x2_to_complex_byte>;
00034
00035 byte_x2_to_complex_byte_sptr make_byte_x2_to_complex_byte();
00036
00037 /*!
00038  * \brief This class adapts two signed char streams
00039  * into a std::complex<signed char> stream
00040  */
00041 class byte_x2_to_complex_byte : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     friend byte_x2_to_complex_byte_sptr make_byte_x2_to_complex_byte();
00050     byte_x2_to_complex_byte();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_BYTE_X2_TO_COMPLEX_BYTE_Hs
```

13.98 complex_byte_to_float_x2.h File Reference

Adapts a std::complex<signed char> stream into two 16-bits (short) streams.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class complex_byte_to_float_x2

  This class adapts a std::complex<signed char> stream into two 16-bits (short) streams.

Typedefs

- using complex_byte_to_float_x2_sptr = gnss_shared_ptr<complex_byte_to_float_x2>

Functions

- complex_byte_to_float_x2_sptr **make_complex_byte_to_float_x2** ()

13.98.1 Detailed Description

Adapts a std::complex<signed char> stream into two 16-bits (short) streams.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file complex_byte_to_float_x2.h.

13.99 complex_byte_to_float_x2.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file complex_byte_to_float_x2.h
00003  * \brief Adapts a std::complex<signed char> stream into two 16-bits (short) streams
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_COMPLEX_BYTE_TO_FLOAT_X2_H
00019 #define GNSS_SDR_COMPLEX_BYTE_TO_FLOAT_X2_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class complex_byte_to_float_x2;
00032
00033 using complex_byte_to_float_x2_sptr = gnss_shared_ptr<complex_byte_to_float_x2>;
00034
00035 complex_byte_to_float_x2_sptr make_complex_byte_to_float_x2();
```

```
00036
00037 /*!
00038  * \brief This class adapts a std::complex<signed char> stream
00039  * into two 16-bits (short) streams
00040  */
00041 class complex_byte_to_float_x2 : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     friend complex_byte_to_float_x2_sptr make_complex_byte_to_float_x2();
00050     complex_byte_to_float_x2();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_COMPLEX_BYTE_TO_FLOAT_X2_H
```

13.100 complex_float_to_complex_byte.h File Reference

Adapts a gr_complex stream into a std::complex<signed char> stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class complex_float_to_complex_byte

  This class adapts a gr_complex stream into a std::complex<signed char> stream.

Typedefs

- using complex_float_to_complex_byte_sptr = gnss_shared_ptr<complex_float_to_complex_byte>

Functions

- complex_float_to_complex_byte_sptr **make_complex_float_to_complex_byte** ()

13.100.1 Detailed Description

Adapts a gr_complex stream into a std::complex<signed char> stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file complex_float_to_complex_byte.h.

13.101 complex_float_to_complex_byte.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file complex_float_to_complex_byte.h
00003  * \brief Adapts a gr_complex stream into a std::complex<signed char> stream
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
```

```
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_COMPLEX_FLOAT_TO_COMPLEX_BYTE_H
00019 #define GNSS_SDR_COMPLEX_FLOAT_TO_COMPLEX_BYTE_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class complex_float_to_complex_byte;
00032
00033 using complex_float_to_complex_byte_sptr = gnss_shared_ptr<complex_float_to_complex_byte>;
00034
00035 complex_float_to_complex_byte_sptr make_complex_float_to_complex_byte();
00036
00037 /*!
00038  * \brief This class adapts a gr_complex stream into a std::complex<signed char> stream
00039  */
00040 class complex_float_to_complex_byte : public gr::sync_block
00041 {
00042 public:
00043     int work(int noutput_items,
00044         gr_vector_const_void_star &input_items,
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     friend complex_float_to_complex_byte_sptr make_complex_float_to_complex_byte();
00049     complex_float_to_complex_byte();
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_COMPLEX_FLOAT_TO_COMPLEX_BYTE_H
```

13.102 conjugate_cc.h File Reference

Conjugate a stream of gr_complex.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class conjugate_cc

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.

Typedefs

- using conjugate_cc_sptr = gnss_shared_ptr<conjugate_cc>

Functions

- conjugate_cc_sptr **make_conjugate_cc** ()

13.102.1 Detailed Description

Conjugate a stream of gr_complex.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file conjugate_cc.h.

13.103 conjugate_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file conjugate_cc.h
00003  * \brief Conjugate a stream of gr_complex
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CONJUGATE_CC_H
00019 #define GNSS_SDR_CONJUGATE_CC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class conjugate_cc;
00032
00033 using conjugate_cc_sptr = gnss_shared_ptr<conjugate_cc>;
00034
00035 conjugate_cc_sptr make_conjugate_cc();
00036
00037 /*!
00038  * \brief This class adapts a std::complex<short> stream
00039  * into two 32-bits (float) streams
00040  */
00041 class conjugate_cc : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     friend conjugate_cc_sptr make_conjugate_cc();
00050     conjugate_cc();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_CONJUGATE_CC_H
```

13.104 conjugate_ic.h File Reference

Conjugate a stream of lv_8sc_t (std::complex<char>).

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class conjugate_ic

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.

Typedefs

- using conjugate_ic_sptr = gnss_shared_ptr<conjugate_ic>

Functions

- conjugate_ic_sptr **make_conjugate_ic** ()

13.104.1 Detailed Description

Conjugate a stream of lv_8sc_t (std::complex<char>).

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file conjugate_ic.h.

13.105 conjugate_ic.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file conjugate_ic.h
00003  * \brief Conjugate a stream of lv_8sc_t ( std::complex<char> )
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CONJUGATE_IC_H
00019 #define GNSS_SDR_CONJUGATE_IC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class conjugate_ic;
00032
00033 using conjugate_ic_sptr = gnss_shared_ptr<conjugate_ic>;
00034
00035 conjugate_ic_sptr make_conjugate_ic();
00036
00037 /*!
00038  * \brief This class adapts a std::complex<short> stream
00039  * into two 32-bits (float) streams
00040  */
00041 class conjugate_ic : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
```

```
00048 private:
00049     friend conjugate_ic_sptr make_conjugate_ic();
00050     conjugate_ic();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_CONJUGATE_IC_H
```

13.106 conjugate_sc.h File Reference

Conjugate a stream of lv_16sc_t (std::complex<short>).

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class conjugate_sc

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.

Typedefs

- using conjugate_sc_sptr = gnss_shared_ptr<conjugate_sc>

Functions

- conjugate_sc_sptr **make_conjugate_sc** ()

13.106.1 Detailed Description

Conjugate a stream of lv_16sc_t (std::complex<short>).

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file conjugate_sc.h.

13.107 conjugate_sc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file conjugate_sc.h
00003  * \brief Conjugate a stream of lv_16sc_t ( std::complex<short> )
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CONJUGATE_SC_H
00019 #define GNSS_SDR_CONJUGATE_SC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
```

```
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class conjugate_sc;
00032
00033 using conjugate_sc_sptr = gnss_shared_ptr<conjugate_sc>;
00034
00035 conjugate_sc_sptr make_conjugate_sc();
00036
00037 /*!
00038  * \brief This class adapts a std::complex<short> stream
00039  * into two 32-bits (float) streams
00040  */
00041 class conjugate_sc : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     friend conjugate_sc_sptr make_conjugate_sc();
00050     conjugate_sc();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_CONJUGATE_SC_H
```

13.108 cshort_to_float_x2.h File Reference

Adapts a std::complex<short> stream into two float streams.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class cshort_to_float_x2

  This class adapts a std::complex<short> stream into two 32-bits (float) streams.

Typedefs

- using cshort_to_float_x2_sptr = gnss_shared_ptr<cshort_to_float_x2>

Functions

- cshort_to_float_x2_sptr **make_cshort_to_float_x2** ()

13.108.1 Detailed Description

Adapts a std::complex<short> stream into two float streams.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cshort_to_float_x2.h.

13.109 cshort_to_float_x2.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cshort_to_float_x2.h
00003  * \brief Adapts a std::complex<short> stream into two float streams
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CSHORT_TO_FLOAT_X2_H
00019 #define GNSS_SDR_CSHORT_TO_FLOAT_X2_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class cshort_to_float_x2;
00032
00033 using cshort_to_float_x2_sptr = gnss_shared_ptr<cshort_to_float_x2>;
00034
00035 cshort_to_float_x2_sptr make_cshort_to_float_x2();
00036
00037 /*!
00038  * \brief This class adapts a std::complex<short> stream
00039  * into two 32-bits (float) streams
00040  */
00041 class cshort_to_float_x2 : public gr::sync_block
00042 {
00043 public:
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     friend cshort_to_float_x2_sptr make_cshort_to_float_x2();
00050     cshort_to_float_x2();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_CSHORT_TO_FLOAT_X2_H
```

13.110 dump_logger_helper.h File Reference

Utility functions for logging to a file.

`#include <fstream>`

Functions

- template<typename T>
  void write_value (std::ofstream &file, T value)

13.110.1 Detailed Description

Utility functions for logging to a file.

Authors

- Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dump_logger_helper.h.

13.110.2 Function Documentation

13.110.2.1 write_value()

```
template<typename T>
void write_value (
            std::ofstream & file,
            T value)
```

Definition at line 25 of file dump_logger_helper.h.

13.111 dump_logger_helper.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dump_logger_helper.h
00003  * \brief Utility functions for logging to a file
00004  * \authors <ul>
00005  *          <li> Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.com
00006  *          </ul>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DUMP_LOGGER_HELPER_H
00020 #define GNSS_SDR_DUMP_LOGGER_HELPER_H
00021
00022 #include <fstream>
00023
00024 template <typename T>
00025 void write_value(std::ofstream& file, T value)
00026 {
00027     file.write(reinterpret_cast<char*>(&value), sizeof(T));
00028 }
00029
00030 #endif  // GNSS_SDR_DUMP_LOGGER_HELPER_H
```

13.112 galileo_e1_signal_replica.h File Reference

This library implements various functions for Galileo E1 signal replica generation.

```
#include <array>
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void galileo_e1_code_gen_sinboc11_float (own::span< float > dest, const std::array< char, 3 > &signal_id, uint32_t prn)

  This function generates Galileo E1 code (can select E1B or E1C sinboc).

- void galileo_e1_code_gen_float_sampled (own::span< float > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)

This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling↩ _freq).

- void galileo_e1_code_gen_float_sampled (own::span< float > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling↩ _freq).

- void galileo_e1_code_gen_complex_sampled (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift, bool secondary_flag)

  This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc and the sample frequency sampling↩ _freq).

- void galileo_e1_code_gen_complex_sampled (own::span< std::complex< float > > dest, const std::array< char, 3 > &signal_id, bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  galileo_e1_code_gen_complex_sampled without secondary_flag for backward compatibility.

13.112.1 Detailed Description

This library implements various functions for Galileo E1 signal replica generation.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_e1_signal_replica.h.

13.113 galileo_e1_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e1_signal_replica.h
00003  * \brief This library implements various functions for Galileo E1 signal
00004  * replica generation
00005  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GALILEO_E1_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GALILEO_E1_SIGNAL_REPLICA_H
00021
00022 #include <array>
00023 #include <complex>
00024 #include <cstdint>
00025 #if HAS_STD_SPAN
00026 #include <span>
00027 namespace own = std;
00028 #else
00029 #include <gsl-lite/gsl-lite.hpp>
00030 namespace own = gsl_lite;
00031 #endif
00032
00033 /** \addtogroup Algorithms_Library
00034  * \{ */
00035 /** \addtogroup Algorithm_libs algorithms_libs
00036  * \{ */
00037
00038
00039 /*!
00040  * \brief This function generates Galileo E1 code (can select E1B or E1C sinboc).
00041  *
00042  */
00043 void galileo_e1_code_gen_sinboc11_float(own::span<float> dest, const std::array<char, 3>& signal_id,
      uint32_t prn);
```

```
00044
00045 /*!
00046  * \brief This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc
00047  * and the sample frequency sampling_freq).
00048  *
00049  */
00050 void galileo_e1_code_gen_float_sampled(own::span<float> dest, const std::array<char, 3>& signal_id,
00051     bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift,
00052     bool secondary_flag);
00053
00054 /*!
00055  * \brief This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc
00056  * and the sample frequency sampling_freq).
00057  *
00058  */
00059 void galileo_e1_code_gen_float_sampled(own::span<float> dest, const std::array<char, 3>& signal_id,
00060     bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift);
00061
00062 /*!
00063  * \brief This function generates Galileo E1 code (can select E1B or E1C, cboc or sinboc
00064  * and the sample frequency sampling_freq).
00065  *
00066  */
00067 void galileo_e1_code_gen_complex_sampled(own::span<std::complex<float» dest, const std::array<char,
      3>& signal_id,
00068     bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift,
00069     bool secondary_flag);
00070
00071 /*!
00072  * \brief galileo_e1_code_gen_complex_sampled without secondary_flag for backward compatibility.
00073  */
00074 void galileo_e1_code_gen_complex_sampled(own::span<std::complex<float» dest, const std::array<char,
      3>& signal_id,
00075     bool cboc, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift);
00076
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_GALILEO_E1_SIGNAL_REPLICA_H
```

13.114 galileo_e5_signal_replica.h File Reference

This library implements various functions for Galileo E5 signal replica generation.

```
#include <array>
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void galileo_e5_a_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn, const std::array< char, 3 > &signal_id)

  Generates Galileo E5a code at 1 sample/chip.

- void galileo_e5_a_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, const std::array< char, 3 > &signal_id, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E5a complex code, shifted to the desired chip and sampled at a frequency sampling_freq.

- void galileo_e5_b_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn, const std::array< char, 3 > &signal_id)

  Generates Galileo E5b code at 1 sample/chip.

- void galileo_e5_b_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, const std::array< char, 3 > &signal_id, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E5b complex code, shifted to the desired chip and sampled at a frequency sampling_freq.

13.114.1 Detailed Description

This library implements various functions for Galileo E5 signal replica generation.

Author

Marc Sales, 2014. marcsales92(at)gmail.com

Piyush Gupta, 2020. piyush04111999@gmail.com

Note

Code added as part of GSoC 2020 Program.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_e5_signal_replica.h.

13.115 galileo_e5_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e5_signal_replica.h
00003  * \brief This library implements various functions for Galileo E5 signal
00004  * replica generation
00005  * \author Marc Sales, 2014. marcsales92(at)gmail.com
00006  * \author Piyush Gupta, 2020. piyush04111999@gmail.com
00007  * \note Code added as part of GSoC 2020 Program.
00008  *
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_GALILEO_E5_SIGNAL_REPLICA_H
00022 #define GNSS_SDR_GALILEO_E5_SIGNAL_REPLICA_H
00023
00024 #include <array>
00025 #include <complex>
00026 #include <cstdint>
00027 #if HAS_STD_SPAN
00028 #include <span>
00029 namespace own = std;
00030 #else
00031 #include <gsl-lite/gsl-lite.hpp>
00032 namespace own = gsl_lite;
00033 #endif
00034
00035 /** \addtogroup Algorithms_Library
00036  * \{ */
00037 /** \addtogroup Algorithm_libs algorithms_libs
00038  * \{ */
00039
00040
00041 /*!
00042  * \brief Generates Galileo E5a code at 1 sample/chip
00043  */
00044 void galileo_e5_a_code_gen_complex_primary(own::span<std::complex<float» dest,
00045     int32_t prn,
00046     const std::array<char, 3>& signal_id);
00047
00048
00049 /*!
00050  * \brief Generates Galileo E5a complex code, shifted to the desired chip and
00051  * sampled at a frequency sampling_freq
00052  */
00053 void galileo_e5_a_code_gen_complex_sampled(own::span<std::complex<float» dest,
00054     uint32_t prn,
00055     const std::array<char, 3>& signal_id,
00056     int32_t sampling_freq,
00057     uint32_t chip_shift);
00058
00059
00060 /*!
00061  * \brief Generates Galileo E5b code at 1 sample/chip
00062  */
00063 void galileo_e5_b_code_gen_complex_primary(own::span<std::complex<float» dest,
00064     int32_t prn,
```

```
00065     const std::array<char, 3>& signal_id);
00066
00067
00068 /*!
00069  * \brief Generates Galileo E5b complex code, shifted to the desired chip and
00070  * sampled at a frequency sampling_freq
00071  */
00072 void galileo_e5_b_code_gen_complex_sampled(own::span<std::complex<float» dest,
00073     uint32_t prn,
00074     const std::array<char, 3>& signal_id,
00075     int32_t sampling_freq,
00076     uint32_t chip_shift);
00077
00078
00079 /** \} */
00080 /** \} */
00081 #endif  // GNSS_SDR_GALILEO_E5_SIGNAL_REPLICA_H
```

13.116 galileo_e6_signal_replica.h File Reference

This library implements various functions for Galileo E6 signal replica generation.

```
#include <array>
#include <complex>
#include <cstdint>
#include <string>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void galileo_e6_b_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6B code at 1 sample/chip.
- void galileo_e6_b_code_gen_float_primary (own::span< float > dest, int32_t prn)

  Generates Galileo E6B code at 1 sample/chip.
- void galileo_e6_b_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E6B complex code, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e6_c_code_gen_complex_primary (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6C codes at 1 sample/chip.
- void galileo_e6_c_code_gen_float_primary (own::span< float > dest, int32_t prn)

  Generates Galileo E6C codes at 1 sample/chip.
- void galileo_e6_c_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates Galileo E6C complex codes, shifted to the desired chip and sampled at a frequency sampling_freq.
- void galileo_e6_c_secondary_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn)

  Generates Galileo E6C secondary codes at 1 sample/chip.
- void galileo_e6_c_secondary_code_gen_float (own::span< float > dest, int32_t prn)

  Generates Galileo E6C secondary codes at 1 sample/chip.
- std::string galileo_e6_c_secondary_code (int32_t prn)

  Generates a string with Galileo E6C secondary codes at 1 sample/chip.

13.116.1 Detailed Description

This library implements various functions for Galileo E6 signal replica generation.

Author

Carles Fernandez-Prades, 2020. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_e6_signal_replica.h.

13.117 galileo_e6_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e6_signal_replica.h
00003  * \brief This library implements various functions for Galileo E6 signal
00004  * replica generation
00005  * \author Carles Fernandez-Prades, 2020. cfernandez(at)cttc.es
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GALILEO_E6_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GALILEO_E6_SIGNAL_REPLICA_H
00021
00022 #include <array>
00023 #include <complex>
00024 #include <cstdint>
00025 #include <string>
00026 #if HAS_STD_SPAN
00027 #include <span>
00028 namespace own = std;
00029 #else
00030 #include <gsl-lite/gsl-lite.hpp>
00031 namespace own = gsl_lite;
00032 #endif
00033
00034 /** \addtogroup Algorithms_Library
00035  * \{ */
00036 /** \addtogroup Algorithm_libs algorithms_libs
00037  * \{ */
00038
00039
00040 /*!
00041  * \brief Generates Galileo E6B code at 1 sample/chip
00042  */
00043 void galileo_e6_b_code_gen_complex_primary(own::span<std::complex<float>> dest,
00044     int32_t prn);
00045
00046
00047 /*!
00048  * \brief Generates Galileo E6B code at 1 sample/chip
00049  */
00050 void galileo_e6_b_code_gen_float_primary(own::span<float> dest, int32_t prn);
00051
00052
00053 /*!
00054  * \brief Generates Galileo E6B complex code, shifted to the desired chip and
00055  * sampled at a frequency sampling_freq
00056  */
00057 void galileo_e6_b_code_gen_complex_sampled(own::span<std::complex<float>> dest,
00058     uint32_t prn,
00059     int32_t sampling_freq,
00060     uint32_t chip_shift);
00061
00062
00063 /*!
00064  * \brief Generates Galileo E6C codes at 1 sample/chip
00065  */
00066 void galileo_e6_c_code_gen_complex_primary(own::span<std::complex<float>> dest,
00067     int32_t prn);
00068
00069
00070 /*!
00071  * \brief Generates Galileo E6C codes at 1 sample/chip
00072  */
00073 void galileo_e6_c_code_gen_float_primary(own::span<float> dest, int32_t prn);
00074
00075
00076 /*!
00077  * \brief Generates Galileo E6C complex codes, shifted to the desired chip and
00078  * sampled at a frequency sampling_freq
00079  */
00080 void galileo_e6_c_code_gen_complex_sampled(own::span<std::complex<float>> dest,
00081     uint32_t prn,
00082     int32_t sampling_freq,
00083     uint32_t chip_shift);
```

```
00084
00085
00086 /*!
00087  * \brief Generates Galileo E6C secondary codes at 1 sample/chip
00088  */
00089 void galileo_e6_c_secondary_code_gen_complex(own::span<std::complex<float» dest,
00090     int32_t prn);
00091
00092
00093 /*!
00094  * \brief Generates Galileo E6C secondary codes at 1 sample/chip
00095  */
00096 void galileo_e6_c_secondary_code_gen_float(own::span<float> dest,
00097     int32_t prn);
00098
00099
00100 /*!
00101  * \brief Generates a string with Galileo E6C secondary codes at 1 sample/chip
00102  */
00103 std::string galileo_e6_c_secondary_code(int32_t prn);
00104
00105
00106 /** \} */
00107 /** \} */
00108 #endif  // GNSS_SDR_GALILEO_E6_SIGNAL_REPLICA_H
```

13.118 geofunctions.h File Reference

A set of coordinate transformations functions and helpers, some of them migrated from MATLAB, for geographic information systems.

```
#include <armadillo>
```

Functions

- arma::mat Skew_symmetric (const arma::vec &a)

  Calculates skew-symmetric matrix.
- double **WGS84_g0** (double Lat_rad)
- double **WGS84_geocentric_radius** (double Lat_geodetic_rad)
- int topocent (double *Az, double *El, double *D, const arma::vec &x, const arma::vec &dx)

  Transformation of vector dx into topocentric coordinate system with origin at x Inputs: x - vector origin coordinates (in ECEF system [X; Y; Z;]) dx - vector ([dX; dY; dZ;]).
- int togeod (double *dphi, double *dlambda, double *h, double a, double finv, double X, double Y, double Z)

  Subroutine to calculate geodetic coordinates latitude, longitude, height given Cartesian coordinates X,Y,Z, and reference ellipsoid values semi-major axis (a) and the inverse of flattening (finv).
- arma::vec Gravity_ECEF (const arma::vec &r_eb_e)

  Calculates acceleration due to gravity resolved about ECEF-frame.
- arma::vec cart2geo (const arma::vec &XYZ, int elipsoid_selection)

  Conversion of Cartesian coordinates (X,Y,Z) to geographical coordinates (latitude, longitude, h) on a selected reference ellipsoid.
- arma::vec **LLH_to_deg** (const arma::vec &LLH)
- double **degtorad** (double angleInDegrees)
- double **radtodeg** (double angleInRadians)
- double **mstoknotsh** (double MetersPerSeconds)
- double **mstokph** (double MetersPerSeconds)
- arma::vec **CTM_to_Euler** (const arma::mat &C)
- arma::mat **Euler_to_CTM** (const arma::vec &eul)
- void **ECEF_to_Geo** (const arma::vec &r_eb_e, const arma::vec &v_eb_e, const arma::mat &C_b_e, arma::↩ vec &LLH, arma::vec &v_eb_n, arma::mat &C_b_n)
- void Geo_to_ECEF (const arma::vec &LLH, const arma::vec &v_eb_n, const arma::mat &C_b_n, arma::vec &r_eb_e, arma::vec &v_eb_e, arma::mat &C_b_e)

  From Geographic to ECEF coordinates.
- void pv_Geo_to_ECEF (double L_b, double lambda_b, double h_b, const arma::vec &v_eb_n, arma::vec &r_eb_e, arma::vec &v_eb_e)

Converts curvilinear to Cartesian position and velocity resolving axes from NED to ECEF This function created 11/↩ 4/2012 by Paul Groves.

- double great_circle_distance (double lat1, double lon1, double lat2, double lon2)

  The Haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes.
- void cart2utm (const arma::vec &r_eb_e, int zone, arma::vec &r_enu)

  Transformation of ECEF (X,Y,Z) to (E,N,U) in UTM, zone 'zone'.
- int findUtmZone (double latitude_deg, double longitude_deg)

  Function finds the UTM zone number for given longitude and latitude.
- double clsin (const arma::colvec &ar, int degree, double argument)

  Clenshaw summation of sinus of argument.
- void clksin (const arma::colvec &ar, int degree, double arg_real, double arg_imag, double *re, double *im)

  Clenshaw summation of sinus with complex argument.

13.118.1 Detailed Description

A set of coordinate transformations functions and helpers, some of them migrated from MATLAB, for geographic information systems.

Author

Javier Arribas, 2018. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file geofunctions.h.

13.119 geofunctions.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file geofunctions.h
00003  * \brief A set of coordinate transformations functions and helpers,
00004  * some of them migrated from MATLAB, for geographic information systems.
00005  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GEOFUNCTIONS_H
00020 #define GNSS_SDR_GEOFUNCTIONS_H
00021
00022 #if ARMA_NO_BOUND_CHECKING
00023 #define ARMA_NO_DEBUG 1
00024 #endif
00025
00026 #include <armadillo>
00027
00028 /** \addtogroup Algorithms_Library
00029  * \{ */
00030 /** \addtogroup Algorithm_libs algorithms_libs
00031  * \{ */
00032
00033
00034 arma::mat Skew_symmetric(const arma::vec &a);  //!< Calculates skew-symmetric matrix
00035
00036 double WGS84_g0(double Lat_rad);
00037
00038 double WGS84_geocentric_radius(double Lat_geodetic_rad);
00039
00040 /*!
00041  * \brief  Transformation of vector dx into topocentric coordinate
```

```
00042  *      system with origin at x
00043  *       Inputs:
00044  *         x           - vector origin coordinates (in ECEF system [X; Y; Z;])
00045  *         dx          - vector ([dX; dY; dZ;]).
00046  *
00047  *        Outputs:
00048  *          D           - vector length. Units like the input
00049  *          Az          - azimuth from north positive clockwise, degrees
00050  *          El          - elevation angle, degrees
00051  *
00052  *          Based on a Matlab function by Kai Borre
00053  */
00054 int topocent(double *Az, double *El, double *D, const arma::vec &x, const arma::vec &dx);
00055
00056 /*!
00057  * \brief Subroutine to calculate geodetic coordinates latitude, longitude,
00058  *   height given Cartesian coordinates X,Y,Z, and reference ellipsoid
00059  *   values semi-major axis (a) and the inverse of flattening (finv).
00060  *
00061  *   The output units of angular quantities will be in decimal degrees
00062  *   (15.5 degrees not 15 deg 30 min). The output units of h will be the
00063  *   same as the units of X,Y,Z,a.
00064  *
00065  *         Inputs:
00066  *             a           - semi-major axis of the reference ellipsoid
00067  *             finv        - inverse of flattening of the reference ellipsoid
00068  *             X,Y,Z       - Cartesian coordinates
00069  *
00070  *         Outputs:
00071  *             dphi        - latitude
00072  *             dlambda     - longitude
00073  *             h           - height above reference ellipsoid
00074  *
00075  *             Based in a Matlab function by Kai Borre
00076  */
00077 int togeod(double *dphi, double *dlambda, double *h, double a, double finv, double X, double Y, double
       Z);
00078
00079 arma::vec Gravity_ECEF(const arma::vec &r_eb_e);  //!< Calculates acceleration due to gravity resolved
       about ECEF-frame
00080
00081 /*!
00082  * \brief Conversion of Cartesian coordinates (X,Y,Z) to geographical
00083  *  coordinates (latitude, longitude, h) on a selected reference ellipsoid.
00084  *
00085  *    Choices of Reference Ellipsoid for Geographical Coordinates
00086  *             0. International Ellipsoid 1924
00087  *             1. International Ellipsoid 1967
00088  *             2. World Geodetic System 1972
00089  *             3. Geodetic Reference System 1980
00090  *             4. World Geodetic System 1984
00091  */
00092 arma::vec cart2geo(const arma::vec &XYZ, int elipsoid_selection);
00093
00094 arma::vec LLH_to_deg(const arma::vec &LLH);
00095
00096 double degtorad(double angleInDegrees);
00097
00098 double radtodeg(double angleInRadians);
00099
00100 double mstoknotsh(double MetersPerSeconds);
00101
00102 double mstokph(double MetersPerSeconds);
00103
00104 arma::vec CTM_to_Euler(const arma::mat &C);
00105
00106 arma::mat Euler_to_CTM(const arma::vec &eul);
00107
00108 void ECEF_to_Geo(const arma::vec &r_eb_e, const arma::vec &v_eb_e, const arma::mat &C_b_e, arma::vec
       &LLH, arma::vec &v_eb_n, arma::mat &C_b_n);
00109
00110
00111 /*!
00112  * \brief From Geographic to ECEF coordinates
00113  *
00114  *  Inputs:
00115  *    LLH           latitude (rad), longitude (rad), height (m)
00116  *    v_eb_n        velocity of body frame w.r.t. ECEF frame, resolved along
00117  *                  north, east, and down (m/s)
00118  *    C_b_n         body-to-NED coordinate transformation matrix
00119  *
00120  *  Outputs:
00121  *    r_eb_e        Cartesian position of body frame w.r.t. ECEF frame, resolved
00122  *                 along ECEF-frame axes (m)
00123  *    v_eb_e        velocity of body frame w.r.t. ECEF frame, resolved along
00124  *                  ECEF-frame axes (m/s)
00125  *    C_b_e         body-to-ECEF-frame coordinate transformation matrix
```

```
00126  *
00127  */
00128 void Geo_to_ECEF(const arma::vec &LLH, const arma::vec &v_eb_n, const arma::mat &C_b_n, arma::vec
      &r_eb_e, arma::vec &v_eb_e, arma::mat &C_b_e);
00129
00130
00131 /*!
00132  * \brief Converts curvilinear to Cartesian position and velocity
00133  * resolving axes from NED to ECEF
00134  * This function created 11/4/2012 by Paul Groves
00135  *
00136  * Inputs:
00137  *    L_b           latitude (rad)
00138  *    lambda_b      longitude (rad)
00139  *    h_b           height (m)
00140  *    v_eb_n        velocity of body frame w.r.t. ECEF frame, resolved along
00141  *                  north, east, and down (m/s)
00142  *
00143  * Outputs:
00144  *    r_eb_e        Cartesian position of body frame w.r.t. ECEF frame, resolved
00145  *                  along ECEF-frame axes (m)
00146  *    v_eb_e        velocity of body frame w.r.t. ECEF frame, resolved along
00147  *                  ECEF-frame axes (m/s)
00148  */
00149 void pv_Geo_to_ECEF(double L_b, double lambda_b, double h_b, const arma::vec &v_eb_n, arma::vec
      &r_eb_e, arma::vec &v_eb_e);
00150
00151
00152 /*!
00153  * \brief The Haversine formula determines the great-circle distance between two points on a sphere
      given their longitudes and latitudes.
00154  */
00155 double great_circle_distance(double lat1, double lon1, double lat2, double lon2);
00156
00157
00158 /*!
00159  * \brief Transformation of ECEF (X,Y,Z) to (E,N,U) in UTM, zone 'zone'.
00160  */
00161 void cart2utm(const arma::vec &r_eb_e, int zone, arma::vec &r_enu);
00162
00163
00164 /*!
00165  * \brief Function finds the UTM zone number for given longitude and latitude.
00166  */
00167 int findUtmZone(double latitude_deg, double longitude_deg);
00168
00169
00170 /*!
00171  * \brief Clenshaw summation of sinus of argument.
00172  */
00173 double clsin(const arma::colvec &ar, int degree, double argument);
00174
00175
00176 /*!
00177  * \brief Clenshaw summation of sinus with complex argument.
00178  */
00179 void clksin(const arma::colvec &ar, int degree, double arg_real, double arg_imag, double *re, double
      *im);
00180
00181
00182 /** \} */
00183 /** \} */
00184 #endif  // GNSS_SDR_GEOFUNCTIONS_H
```

13.120 glonass_l1_signal_replica.h

```
00001 /*!
00002  * \file glonass_l1_signal_processing.h
00003  * \brief This file implements various functions for GLONASS L1 CA signal
00004  * replica generation
00005  * \author Gabriel Araujo, 2017. gabriel.araujo(at)ieee.org
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
```

```
00019 #ifndef GNSS_SDR_GLONASS_L1_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GLONASS_L1_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #if HAS_STD_SPAN
00025 #include <span>
00026 namespace own = std;
00027 #else
00028 #include <gsl-lite/gsl-lite.hpp>
00029 namespace own = gsl_lite;
00030 #endif
00031
00032 /** \addtogroup Algorithms_Library
00033  * \{ */
00034 /** \addtogroup Algorithm_libs algorithms_libs
00035  * \{ */
00036
00037
00038 //! Generates complex GLONASS L1 C/A code for the desired SV ID and code shift
00039 void glonass_l1_ca_code_gen_complex(own::span<std::complex<float» dest, uint32_t chip_shift);
00040
00041 //! Generates float GLONASS L1 C/A code for the desired SV ID and code shift
00042 void glonass_l1_ca_code_gen_float(own::span<float> dest, uint32_t chip_shift);
00043
00044 //! Generates complex GLONASS L1 C/A code for the desired SV ID and code shift, and sampled to
      specific sampling frequency
00045 void glonass_l1_ca_code_gen_complex_sampled(own::span<std::complex<float» dest, int32_t sampling_freq,
      uint32_t chip_shift);
00046
00047
00048 /** \} */
00049 /** \} */
00050 #endif  // GNSS_SDR_GLONASS_L1_SIGNAL_REPLICA_H
```

13.121 glonass_l2_signal_replica.h File Reference

This file implements various functions for GLONASS L2 CA signal replica generation.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void glonass_l2_ca_code_gen_complex (own::span< std::complex< float > > dest, uint32_t chip_shift)

  Generates complex GLONASS L2 C/A code for the desired SV ID and code shift.
- void glonass_l2_ca_code_gen_float (own::span< float > dest, uint32_t chip_shift)

  Generates float GLONASS L2 C/A code for the desired SV ID and code shift.
- void glonass_l2_ca_code_gen_complex_sampled (own::span< std::complex< float > > dest, int32_t sampling_freq, uint32_t chip_shift)

  Generates complex GLONASS L2 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.

13.121.1 Detailed Description

This file implements various functions for GLONASS L2 CA signal replica generation.

Author

Damian Miralles, 2018, dmiralles2009(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_l2_signal_replica.h.

13.122 glonass_l2_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_l2_signal_replica.h
00003  * \brief This file implements various functions for GLONASS L2 CA signal
00004  * replica generation
00005  * \author Damian Miralles, 2018, dmiralles2009(at)gmail.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GLONASS_L2_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GLONASS_L2_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #if HAS_STD_SPAN
00025 #include <span>
00026 namespace own = std;
00027 #else
00028 #include <gsl-lite/gsl-lite.hpp>
00029 namespace own = gsl_lite;
00030 #endif
00031
00032 /** \addtogroup Algorithms_Library
00033  * \{ */
00034 /** \addtogroup Algorithm_libs algorithms_libs
00035  * \{ */
00036
00037
00038 //! Generates complex GLONASS L2 C/A code for the desired SV ID and code shift
00039 void glonass_l2_ca_code_gen_complex(own::span<std::complex<float>> dest, uint32_t chip_shift);
00040
00041 //! Generates float GLONASS L2 C/A code for the desired SV ID and code shift
00042 void glonass_l2_ca_code_gen_float(own::span<float> dest, uint32_t chip_shift);
00043
00044 //! Generates complex GLONASS L2 C/A code for the desired SV ID and code shift, and sampled to
      specific sampling frequency
00045 void glonass_l2_ca_code_gen_complex_sampled(own::span<std::complex<float>> dest, int32_t sampling_freq,
      uint32_t chip_shift);
00046
00047
00048 /** \} */
00049 /** \} */
00050 #endif  // GNSS_SDR_GLONASS_L2_SIGNAL_REPLICA_H
```

13.123 gnss_circular_deque.h File Reference

This class implements a circular deque for Gnss_Synchro.

```
#include <boost/circular_buffer.hpp>
#include <vector>
```

Classes

- class Gnss_circular_deque< T >

13.123.1 Detailed Description

This class implements a circular deque for Gnss_Synchro.

Author

Antonio Ramos, 2018. antonio.ramosdet(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_circular_deque.h.

13.124 gnss_circular_deque.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_circular_deque.h
00003  * \brief This class implements a circular deque for Gnss_Synchro
00004  * \author Antonio Ramos, 2018. antonio.ramosdet(at)gmail.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CIRCULAR_DEQUE_H
00019 #define GNSS_SDR_CIRCULAR_DEQUE_H
00020
00021 #include <boost/circular_buffer.hpp>
00022 #include <vector>
00023
00024 /** \addtogroup Algorithms_Library
00025  * \{ */
00026 /** \addtogroup Algorithm_libs algorithms_libs
00027  * \{ */
00028
00029
00030 template <class T>
00031 class Gnss_circular_deque
00032 {
00033 public:
00034     Gnss_circular_deque();                                          //!< Default constructor
00035     Gnss_circular_deque(unsigned int max_size, unsigned int nchann);  //!< nchann = number of
      channels; max_size = channel capacity
00036     unsigned int size(unsigned int ch) const;                       //!< Returns the number of
      available elements in a channel
00037     T& at(unsigned int ch, unsigned int pos);                       //!< Returns a reference to an
      element with bound checking
00038     const T& get(unsigned int ch, unsigned int pos) const;          //!< Returns a const reference
      to an element without bound checking
00039     T& front(unsigned int ch);                                      //!< Returns a reference to the
      first element in the deque
00040     T& back(unsigned int ch);                                       //!< Returns a reference to the
      last element in the deque
00041     void push_back(unsigned int ch, const T& new_data);             //!< Inserts an element at the
      end of the deque
00042     void pop_front(unsigned int ch);                                //!< Removes the first element
      of the deque
00043     void clear(unsigned int ch);                                    //!< Removes all the elements of
      the deque (Sets size to 0). Capacity is not modified
00044     void reset(unsigned int max_size, unsigned int nchann);         //!< Removes all the elements in
      all the channels. Re-sets the number of channels and their capacity
00045     void reset();                                                   //!< Removes all the channels
      (Sets nchann to 0)
00046
00047 private:
00048     std::vector<boost::circular_buffer<T» d_data;
00049 };
00050
00051
00052 template <class T>
00053 Gnss_circular_deque<T>::Gnss_circular_deque()
00054 {
00055     reset();
00056 }
00057
00058
00059 template <class T>
00060 Gnss_circular_deque<T>::Gnss_circular_deque(unsigned int max_size, unsigned int nchann)
00061 {
00062     reset(max_size, nchann);
00063 }
00064
00065
00066 template <class T>
00067 unsigned int Gnss_circular_deque<T>::size(unsigned int ch) const
00068 {
00069     return d_data[ch].size();
00070 }
00071
00072
```

```
00073 template <class T>
00074 T& Gnss_circular_deque<T>::back(unsigned int ch)
00075 {
00076     return d_data[ch].back();
00077 }
00078
00079
00080 template <class T>
00081 T& Gnss_circular_deque<T>::front(unsigned int ch)
00082 {
00083     return d_data[ch].front();
00084 }
00085
00086
00087 template <class T>
00088 T& Gnss_circular_deque<T>::at(unsigned int ch, unsigned int pos)
00089 {
00090     return d_data.at(ch).at(pos);
00091 }
00092
00093
00094 template <class T>
00095 const T& Gnss_circular_deque<T>::get(unsigned int ch, unsigned int pos) const
00096 {
00097     return d_data[ch][pos];
00098 }
00099
00100
00101 template <class T>
00102 void Gnss_circular_deque<T>::clear(unsigned int ch)
00103 {
00104     d_data[ch].clear();
00105 }
00106
00107
00108 template <class T>
00109 void Gnss_circular_deque<T>::reset(unsigned int max_size, unsigned int nchann)
00110 {
00111     d_data.clear();
00112     if (max_size > 0 && nchann > 0)
00113         {
00114             for (unsigned int i = 0; i < nchann; i++)
00115                 {
00116                     d_data.push_back(boost::circular_buffer<T>(max_size));
00117                 }
00118         }
00119 }
00120
00121
00122 template <class T>
00123 void Gnss_circular_deque<T>::reset()
00124 {
00125     d_data.clear();
00126 }
00127
00128
00129 template <class T>
00130 void Gnss_circular_deque<T>::pop_front(unsigned int ch)
00131 {
00132     d_data[ch].pop_front();
00133 }
00134
00135
00136 template <class T>
00137 void Gnss_circular_deque<T>::push_back(unsigned int ch, const T& new_data)
00138 {
00139     d_data[ch].push_back(new_data);
00140 }
00141
00142
00143 /** \} */
00144 /** \} */
00145 #endif  // GNSS_SDR_CIRCULAR_DEQUE_H
```

13.125 gnss_sdr_create_directory.h File Reference

Create a directory.

`#include <string>`

Functions

- bool **gnss_sdr_create_directory** (const std::string &foldername)

13.125.1 Detailed Description

Create a directory.

Author

Carles Fernandez-Prades, 2018. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_create_directory.h.

13.126 gnss_sdr_create_directory.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_create_directory.h
00003  * \brief Create a directory
00004  * \author Carles Fernandez-Prades, 2018. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_CREATE_DIRECTORY_H
00019 #define GNSS_SDR_GNSS_SDR_CREATE_DIRECTORY_H
00020
00021 #include <string>
00022
00023 /** \addtogroup Algorithms_Library
00024  * \{ */
00025 /** \addtogroup Algorithm_libs algorithms_libs
00026  * \{ */
00027
00028
00029 bool gnss_sdr_create_directory(const std::string& foldername);
00030
00031
00032 /** \} */
00033 /** \} */
00034 #endif  // GNSS_SDR_GNSS_SDR_CREATE_DIRECTORY_H
```

13.127 gnss_sdr_fft.h File Reference

Helper file for FFT interface.

```
#include "gnss_sdr_make_unique.h"
#include <gnuradio/fft/fft.h>
#include <memory>
#include <utility>
```

Typedefs

- using gnss_fft_complex_fwd = gr::fft::fft_complex
- using gnss_fft_complex_rev = gr::fft::fft_complex
- template<typename T>
  using gnss_fft_fwd_unique_ptr = std::unique_ptr<T>
- template<typename T>
  using gnss_fft_rev_unique_ptr = std::unique_ptr<T>

Functions

- template<typename... Args>
  gnss_fft_fwd_unique_ptr< gr::fft::fft_complex > gnss_fft_fwd_make_unique (Args &&... args)
- template<typename... Args>
  gnss_fft_rev_unique_ptr< gr::fft::fft_complex > gnss_fft_rev_make_unique (Args &&... args)

13.127.1 Detailed Description

Helper file for FFT interface.

Author

Carles Fernandez Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_fft.h.

13.127.2 Typedef Documentation

13.127.2.1 gnss_fft_complex_fwd

```
using gnss_fft_complex_fwd = gr::fft::fft_complex
```
Definition at line 46 of file gnss_sdr_fft.h.

13.127.2.2 gnss_fft_complex_rev

```
using gnss_fft_complex_rev = gr::fft::fft_complex
```
Definition at line 47 of file gnss_sdr_fft.h.

13.127.2.3 gnss_fft_fwd_unique_ptr

```
template<typename T>
using gnss_fft_fwd_unique_ptr = std::unique_ptr<T>
```
Definition at line 49 of file gnss_sdr_fft.h.

13.127.2.4 gnss_fft_rev_unique_ptr

```
template<typename T>
using gnss_fft_rev_unique_ptr = std::unique_ptr<T>
```
Definition at line 56 of file gnss_sdr_fft.h.

13.127.3 Function Documentation

13.127.3.1 gnss_fft_fwd_make_unique()

```
template<typename... Args>
gnss_fft_fwd_unique_ptr< gr::fft::fft_complex > gnss_fft_fwd_make_unique (
            Args &&... args)
```
Definition at line 51 of file gnss_sdr_fft.h.

13.127.3.2 gnss_fft_rev_make_unique()

```
template<typename... Args>
gnss_fft_rev_unique_ptr< gr::fft::fft_complex > gnss_fft_rev_make_unique (
            Args &&... args)
```
Definition at line 58 of file gnss_sdr_fft.h.

13.128 gnss_sdr_fft.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_fft.h
00003  * \brief Helper file for FFT interface
00004  * \author Carles Fernandez Prades, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_FFT_H
00019 #define GNSS_SDR_GNSS_SDR_FFT_H
00020
00021 #include "gnss_sdr_make_unique.h"
00022 #include <gnuradio/fft/fft.h>
00023 #include <memory>
00024 #include <utility>
00025
00026 #if GNURADIO_FFT_USES_TEMPLATES
00027 using gnss_fft_complex_fwd = gr::fft::fft_complex_fwd;
00028 using gnss_fft_complex_rev = gr::fft::fft_complex_rev;
00029 template <typename T>
00030 using gnss_fft_fwd_unique_ptr = std::unique_ptr<T>;
00031 template <typename... Args>
00032 gnss_fft_fwd_unique_ptr<gr::fft::fft_complex_fwd> gnss_fft_fwd_make_unique(Args&&... args)
00033 {
00034     return std::make_unique<gr::fft::fft_complex_fwd>(std::forward<Args>(args)...);
00035 }
00036 template <typename T>
00037 using gnss_fft_rev_unique_ptr = std::unique_ptr<T>;
00038 template <typename... Args>
00039 gnss_fft_rev_unique_ptr<gr::fft::fft_complex_rev> gnss_fft_rev_make_unique(Args&&... args)
00040 {
00041     return std::make_unique<gr::fft::fft_complex_rev>(std::forward<Args>(args)...);
00042 }
00043
00044 #else
00045
00046 using gnss_fft_complex_fwd = gr::fft::fft_complex;
00047 using gnss_fft_complex_rev = gr::fft::fft_complex;
00048 template <typename T>
00049 using gnss_fft_fwd_unique_ptr = std::unique_ptr<T>;
00050 template <typename... Args>
00051 gnss_fft_fwd_unique_ptr<gr::fft::fft_complex> gnss_fft_fwd_make_unique(Args&&... args)
00052 {
00053     return std::make_unique<gr::fft::fft_complex>(std::forward<Args>(args)..., true);
00054 }
00055 template <typename T>
00056 using gnss_fft_rev_unique_ptr = std::unique_ptr<T>;
00057 template <typename... Args>
00058 gnss_fft_rev_unique_ptr<gr::fft::fft_complex> gnss_fft_rev_make_unique(Args&&... args)
00059 {
00060     return std::make_unique<gr::fft::fft_complex>(std::forward<Args>(args)..., false);
00061 }
00062
00063 #endif
00064
00065 #endif  // GNSS_SDR_GNSS_SDR_FFT_H
```

13.129 gnss_sdr_filesystem.h File Reference

Helper file for fylesystem library interface.

```
#include <boost/filesystem/operations.hpp>
#include <boost/filesystem/path.hpp>
#include <boost/filesystem/path_traits.hpp>
#include <boost/system/error_code.hpp>
```

13.129.1 Detailed Description

Helper file for fylesystem library interface.

Author

Carles Fernandez Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_filesystem.h.

13.130 gnss_sdr_filesystem.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_filesystem.h
00003  * \brief Helper file for fylesystem library interface
00004  * \author Carles Fernandez Prades, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_FILESYSTEM_H
00019 #define GNSS_SDR_GNSS_SDR_FILESYSTEM_H
00020
00021 // clang-format off
00022 #if HAS_STD_FILESYSTEM
00023 #include <system_error>
00024 namespace errorlib = std;
00025 #if HAS_STD_FILESYSTEM_EXPERIMENTAL
00026 #include <experimental/filesystem>
00027 namespace fs = std::experimental::filesystem;
00028 #else
00029 #include <filesystem>
00030 namespace fs = std::filesystem;
00031 #endif
00032 #else
00033 #include <boost/filesystem/operations.hpp>   // for create_directories, exists
00034 #include <boost/filesystem/path.hpp>         // for path, operator«
00035 #include <boost/filesystem/path_traits.hpp>  // for filesystem
00036 #include <boost/system/error_code.hpp>       // for error_code
00037 namespace fs = boost::filesystem;
00038 namespace errorlib = boost::system;
00039 #endif
00040 // clang-format on
00041
00042
00043 #endif  // GNSS_SDR_GNSS_SDR_FILESYSTEM_H
```

13.131 gnss_sdr_flags.h File Reference

Helper file for gnss-sdr commandline flags.

```
#include <cstdint>
#include <string>
#include <absl/flags/declare.h>
#include <absl/flags/flag.h>
#include <cstdlib>
#include <iostream>
#include <sys/stat.h>
#include <vector>
```

Functions

- ABSL_DECLARE_FLAG (std::string, c)

  Path to the configuration file.
- ABSL_DECLARE_FLAG (std::string, config_file)

  Path to the configuration file.
- ABSL_DECLARE_FLAG (std::string, log_dir)

  Path to the folder in which logging will be stored.
- ABSL_DECLARE_FLAG (std::string, s)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (std::string, signal_source)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (std::string, timestamp_source)

  Path to the file containing the signal samples.
- ABSL_DECLARE_FLAG (bool, rf_shutdown)

  Shutdown RF when program exits.
- ABSL_DECLARE_FLAG (int32_t, doppler_max)

  If defined, maximum Doppler value in the search grid, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, doppler_step)

  If defined, sets the frequency step in the search grid, in Hz, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, cn0_samples)

  Number of correlator outputs used for CN0 estimation.
- ABSL_DECLARE_FLAG (int32_t, cn0_min)

  Minimum valid CN0 (in dB-Hz).
- ABSL_DECLARE_FLAG (int32_t, max_lock_fail)

  Maximum number of code lock failures before dropping a satellite.
- ABSL_DECLARE_FLAG (int32_t, max_carrier_lock_fail)

  Maximum number of carrier lock failures before dropping a satellite.
- ABSL_DECLARE_FLAG (double, carrier_lock_th)

  Carrier lock threshold (in rad).
- ABSL_DECLARE_FLAG (double, dll_bw_hz)

  Bandwidth of the DLL low pass filter, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (double, pll_bw_hz)

  Bandwidth of the PLL low pass filter, in Hz (overrides the configuration file).
- ABSL_DECLARE_FLAG (int32_t, carrier_smoothing_factor)

  Sets carrier smoothing factor M (overrides the configuration file).
- ABSL_DECLARE_FLAG (std::string, RINEX_version)

  If defined, specifies the RINEX version (2.11 or 3.02). Overrides the configuration file.
- ABSL_DECLARE_FLAG (std::string, RINEX_name)

  If defined, specifies the RINEX files base name.
- ABSL_DECLARE_FLAG (bool, keyboard)

  If set to false, disables the keyboard listener. Only for debug purposes (e.g. ASAN mode termination).
- bool **ValidateFlags** ()

Variables

- const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200

13.131.1 Detailed Description

Helper file for gnss-sdr commandline flags.

Author

Carles Fernandez-Prades, 2018-2024. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_flags.h.

13.132 gnss_sdr_flags.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_flags.h
00003  * \brief Helper file for gnss-sdr commandline flags
00004  * \author Carles Fernandez-Prades, 2018-2024. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_FLAGS_H
00019 #define GNSS_SDR_GNSS_SDR_FLAGS_H
00020
00021 #include <cstdint>
00022 #include <string>
00023
00024 #if USE_GLOG_AND_GFLAGS
00025 #include <gflags/gflags.h>
00026 #else
00027 #include <absl/flags/declare.h>
00028 #include <absl/flags/flag.h>
00029 #include <cstdlib>
00030 #include <iostream>
00031 #include <sys/stat.h>
00032 #include <vector>
00033 #endif
00034
00035
00036 /** \addtogroup Algorithms_Library
00037  * \{ */
00038 /** \addtogroup Gflags gnss_sdr_flags
00039  * Library for command-line handling.
00040  * \{ */
00041
00042 #if USE_GLOG_AND_GFLAGS
00043 DECLARE_string(c);            //!< Path to the configuration file.
00044 DECLARE_string(config_file);  //!< Path to the configuration file.
00045
00046 DECLARE_string(log_dir);  //!< Path to the folder in which logging will be stored.
00047
00048 // Declare flags for signal sources
00049 DECLARE_string(s);                //!< Path to the file containing the signal samples.
00050 DECLARE_string(signal_source);    //!< Path to the file containing the signal samples.
00051 DECLARE_string(timestamp_source); //!< Path to the file containing the signal samples.
00052 DECLARE_bool(rf_shutdown);        //!< Shutdown RF when program exits.
00053
00054 // Declare flags for acquisition blocks
00055 DECLARE_int32(doppler_max);   //!< If defined, maximum Doppler value in the search grid, in Hz
      (overrides the configuration file).
00056 DECLARE_int32(doppler_step);  //!< If defined, sets the frequency step in the search grid, in Hz, in
      Hz (overrides the configuration file).
00057
00058 // Declare flags for tracking blocks
00059 DECLARE_int32(cn0_samples);           //!< Number of correlator outputs used for CN0 estimation.
00060 DECLARE_int32(cn0_min);               //!< Minimum valid CN0 (in dB-Hz).
00061 DECLARE_int32(max_lock_fail);         //!< Maximum number of code lock failures before dropping a
      satellite.
00062 DECLARE_int32(max_carrier_lock_fail); //!< Maximum number of carrier lock failures before dropping a
      satellite.
00063 DECLARE_double(carrier_lock_th);      //!< Carrier lock threshold (in rad).
```

```
00064 DECLARE_double(dll_bw_hz);             //!< Bandwidth of the DLL low pass filter, in Hz (overrides the
      configuration file).
00065 DECLARE_double(pll_bw_hz);             //!< Bandwidth of the PLL low pass filter, in Hz (overrides the
      configuration file).
00066
00067 // Declare flags for observables block
00068 DECLARE_int32(carrier_smoothing_factor);  //!< Sets carrier smoothing factor M (overrides the
      configuration file).
00069 const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200;
00070
00071 // Declare flags for PVT
00072 DECLARE_string(RINEX_version);  //!< If defined, specifies the RINEX version (2.11 or 3.02). Overrides
      the configuration file.
00073 DECLARE_string(RINEX_name);     //!< If defined, specifies the RINEX files base name
00074 DECLARE_bool(keyboard);         //!< If set to false, disables the keyboard listener. Only for debug
      purposes (e.g. ASAN mode termination)
00075
00076 #else
00077 ABSL_DECLARE_FLAG(std::string, c);            //!< Path to the configuration file.
00078 ABSL_DECLARE_FLAG(std::string, config_file);  //!< Path to the configuration file.
00079
00080 ABSL_DECLARE_FLAG(std::string, log_dir);  //!< Path to the folder in which logging will be stored.
00081
00082 // Declare flags for signal sources
00083 ABSL_DECLARE_FLAG(std::string, s);                //!< Path to the file containing the signal
      samples.
00084 ABSL_DECLARE_FLAG(std::string, signal_source);    //!< Path to the file containing the signal
      samples.
00085 ABSL_DECLARE_FLAG(std::string, timestamp_source);  //!< Path to the file containing the signal
      samples.
00086 ABSL_DECLARE_FLAG(bool, rf_shutdown);             //!< Shutdown RF when program exits.
00087
00088 // Declare flags for acquisition blocks
00089 ABSL_DECLARE_FLAG(int32_t, doppler_max);   //!< If defined, maximum Doppler value in the search grid,
      in Hz (overrides the configuration file).
00090 ABSL_DECLARE_FLAG(int32_t, doppler_step);  //!< If defined, sets the frequency step in the search
      grid, in Hz, in Hz (overrides the configuration file).
00091
00092 // Declare flags for tracking blocks
00093 ABSL_DECLARE_FLAG(int32_t, cn0_samples);           //!< Number of correlator outputs used for CN0
      estimation.
00094 ABSL_DECLARE_FLAG(int32_t, cn0_min);               //!< Minimum valid CN0 (in dB-Hz).
00095 ABSL_DECLARE_FLAG(int32_t, max_lock_fail);         //!< Maximum number of code lock failures before
      dropping a satellite.
00096 ABSL_DECLARE_FLAG(int32_t, max_carrier_lock_fail);  //!< Maximum number of carrier lock failures
      before dropping a satellite.
00097 ABSL_DECLARE_FLAG(double, carrier_lock_th);        //!< Carrier lock threshold (in rad).
00098 ABSL_DECLARE_FLAG(double, dll_bw_hz);              //!< Bandwidth of the DLL low pass filter, in Hz
      (overrides the configuration file).
00099 ABSL_DECLARE_FLAG(double, pll_bw_hz);              //!< Bandwidth of the PLL low pass filter, in Hz
      (overrides the configuration file).
00100
00101 // Declare flags for observables block
00102 ABSL_DECLARE_FLAG(int32_t, carrier_smoothing_factor);  //!< Sets carrier smoothing factor M (overrides
      the configuration file).
00103 const int32_t DEFAULT_CARRIER_SMOOTHING_FACTOR = 200;
00104
00105 // Declare flags for PVT
00106 ABSL_DECLARE_FLAG(std::string, RINEX_version);  //!< If defined, specifies the RINEX version (2.11 or
      3.02). Overrides the configuration file.
00107 ABSL_DECLARE_FLAG(std::string, RINEX_name);     //!< If defined, specifies the RINEX files base name
00108 ABSL_DECLARE_FLAG(bool, keyboard);              //!< If set to false, disables the keyboard listener.
      Only for debug purposes (e.g. ASAN mode termination)
00109
00110 static inline void GetTempDirectories(std::vector<std::string>& list)
00111 {
00112     list.clear();
00113     // Directories, in order of preference. If we find a dir that
00114     // exists, we stop adding other less-preferred dirs
00115     const char* candidates[] = {
00116         // Non-null only during unittest/regtest
00117         std::getenv("TEST_TMPDIR"),
00118
00119         // Explicitly-supplied temp dirs
00120         std::getenv("TMPDIR"),
00121         std::getenv("TMP"),
00122
00123         // If all else fails
00124         "/tmp",
00125     };
00126     for (auto d : candidates)
00127         {
00128             if (!d) continue;  // Empty env var
00129
00130             // Make sure we don't surprise anyone who's expecting a '/'
00131             std::string dstr = d;
00132             if (dstr[dstr.size() - 1] != '/')
```

```
00133                 {
00134                     dstr += "/";
00135                 }
00136             list.push_back(dstr);
00137 
00138             struct stat statbuf;
00139             if (!stat(d, &statbuf) && S_ISDIR(statbuf.st_mode))
00140                 {
00141                     // We found a dir that exists - we're done.
00142                     return;
00143                 }
00144         }
00145 }
00146 
00147 
00148 static inline void GetExistingTempDirectories(std::vector<std::string>& list)
00149 {
00150     GetTempDirectories(list);
00151     auto i_dir = list.begin();
00152     while (i_dir != list.end())
00153         {
00154             if (access(i_dir->c_str(), 0))
00155                 {
00156                     i_dir = list.erase(i_dir);
00157                 }
00158             else
00159                 {
00160                     ++i_dir;
00161                 }
00162         };
00163 }
00164 
00165 
00166 static inline std::string GetTempDir()
00167 {
00168     std::vector<std::string> temp_directories_list;
00169     GetExistingTempDirectories(temp_directories_list);
00170 
00171     if (temp_directories_list.empty())
00172         {
00173             std::cerr « "No temporary directory found\n";
00174             exit(EXIT_FAILURE);
00175         }
00176 
00177     // Use first directory from list of existing temporary directories.
00178     return temp_directories_list.front();
00179 }
00180 
00181 bool ValidateFlags();
00182 
00183 #endif
00184 
00185 /** \} */
00186 /** \} */
00187 #endif  // GNSS_SDR_GNSS_SDR_FLAGS_H
```

13.133 gnss_sdr_make_unique.h File Reference

This file implements std::make_unique for C++11.

13.133.1 Detailed Description

This file implements std::make_unique for C++11.

Author

Carles Fernandez-Prades, 2020. cfernandez(at)cttc.es

Based on https://stackoverflow.com/a/17902439
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_make_unique.h.

13.134 gnss_sdr_make_unique.h

Go to the documentation of this file.

```
00001 /*!
```

```
00002  * \file gnss_sdr_make_unique.h
00003  * \brief This file implements std::make_unique for C++11
00004  *
00005  * \author Carles Fernandez-Prades, 2020. cfernandez(at)cttc.es
00006  *
00007  * Based on https://stackoverflow.com/a/17902439
00008  *
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_GNSS_SDR_MAKE_UNIQUE_H
00022 #define GNSS_SDR_GNSS_SDR_MAKE_UNIQUE_H
00023
00024 #if __cplusplus == 201103L
00025
00026 #include <cstddef>
00027 #include <memory>
00028 #include <type_traits>
00029 #include <utility>
00030
00031 /** \addtogroup Algorithms_Library
00032  * \{ */
00033 /** \addtogroup Algorithm_libs algorithms_libs
00034  * \{ */
00035
00036
00037 namespace std
00038 {
00039 template <class T>
00040 struct _Unique_if
00041 {
00042     typedef unique_ptr<T> _Single_object;
00043 };
00044
00045 template <class T>
00046 struct _Unique_if<T[]>
00047 {
00048     typedef unique_ptr<T[]> _Unknown_bound;
00049 };
00050
00051 template <class T, size_t N>
00052 struct _Unique_if<T[N]>
00053 {
00054     typedef void _Known_bound;
00055 };
00056
00057 template <class T, class... Args>
00058 typename _Unique_if<T>::_Single_object
00059 make_unique(Args&&... args)
00060 {
00061     return unique_ptr<T>(new T(std::forward<Args>(args)...));
00062 }
00063
00064 template <class T>
00065 typename _Unique_if<T>::_Unknown_bound
00066 make_unique(size_t n)
00067 {
00068     typedef typename remove_extent<T>::type U;
00069     return unique_ptr<T>(new U[n]());
00070 }
00071
00072 template <class T, class... Args>
00073 typename _Unique_if<T>::_Known_bound
00074 make_unique(Args&&...) = delete;
00075 }  // namespace std
00076
00077 #endif  // __cplusplus == 201103L
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_GNSS_SDR_MAKE_UNIQUE_H
```

13.135 gnss_sdr_string_literals.h File Reference

This file implements the ""s operator for std::string in C++11, and puts it into the std::string_literals namespace. This is already implemented in C++14, so this is only compiled when using C++11. The .cc file is required for avoiding the duplication of symbols.

13.135.1 Detailed Description

This file implements the ""s operator for std::string in C++11, and puts it into the std::string_literals namespace. This is already implemented in C++14, so this is only compiled when using C++11. The .cc file is required for avoiding the duplication of symbols.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_string_literals.h.

13.136 gnss_sdr_string_literals.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_string_literals.h
00003  * \brief This file implements the ""s operator for std::string in C++11, and
00004  * puts it into the std::string_literals namespace. This is already implemented
00005  * in C++14, so this is only compiled when using C++11. The .cc file is required
00006  * for avoiding the duplication of symbols.
00007  *
00008  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00009  *
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_STRING_LITERALS_H
00023 #define GNSS_SDR_STRING_LITERALS_H
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030 #if __cplusplus == 201103L
00031
00032 #include <cstddef>
00033 #include <string>
00034
00035 namespace std
00036 {
00037 namespace string_literals
00038 {
00039 std::string operator"" s(const char* str, std::size_t len);
00040 }  // namespace string_literals
00041 }  // namespace std
00042
00043 #endif  // __cplusplus == 201103L
00044
00045 /** \} */
00046 /** \} */
00047
00048 #endif  // GNSS_SDR_STRING_LITERALS_H
```

13.137 gnss_signal_replica.h File Reference

This library gathers a few functions used for GNSS signal replica generation regardless of system used.

```
#include <complex>
#include <cstdint>
#include <string>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void complex_exp_gen (own::span< std::complex< float > > dest, double freq, double sampling_freq)

  This function generates a complex exponential in dest.
- void complex_exp_gen_conj (own::span< std::complex< float > > dest, double freq, double sampling_freq)

  This function generates a conjugate complex exponential in dest.
- void hex_to_binary_converter (own::span< int32_t > dest, char from)

  This function makes a conversion from hex (the input is a char) to binary (the output are 4 ints with +1 or -1 values).
- std::string hex_to_binary_string (char from)

  This function makes a conversion from hex (the input is a char) to binary (the output is a string of 4 char with 0 or 1 values).
- void resampler (const own::span< float > from, own::span< float > dest, float fs_in, float fs_out)

  This function resamples a sequence of float values.
- void resampler (own::span< const std::complex< float > > from, own::span< std::complex< float > > dest, float fs_in, float fs_out)

  This function resamples a sequence of complex values.

13.137.1 Detailed Description

This library gathers a few functions used for GNSS signal replica generation regardless of system used.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_signal_replica.h.

13.138 gnss_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_signal_replica.h
00003  * \brief This library gathers a few functions used for GNSS signal replica
00004  * generation regardless of system used
00005  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GNSS_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GNSS_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #include <string>
00025 #if HAS_STD_SPAN
00026 #include <span>
00027 namespace own = std;
00028 #else
00029 #include <gsl-lite/gsl-lite.hpp>
```

```
00030 namespace own = gsl_lite;
00031 #endif
00032
00033 /** \addtogroup Algorithms_Library
00034  * \{ */
00035 /** \addtogroup Algorithm_libs algorithms_libs
00036  * \{ */
00037
00038
00039 /*!
00040  * \brief This function generates a complex exponential in dest.
00041  *
00042  */
00043 void complex_exp_gen(own::span<std::complex<float» dest, double freq, double sampling_freq);
00044
00045 /*!
00046  * \brief This function generates a conjugate complex exponential in dest.
00047  *
00048  */
00049 void complex_exp_gen_conj(own::span<std::complex<float» dest, double freq, double sampling_freq);
00050
00051 /*!
00052  * \brief This function makes a conversion from hex (the input is a char)
00053  *  to binary (the output are 4 ints with +1 or -1 values).
00054  *
00055  */
00056 void hex_to_binary_converter(own::span<int32_t> dest, char from);
00057
00058 /*!
00059  * \brief This function makes a conversion from hex (the input is a char)
00060  *  to binary (the output is a string of 4 char with 0 or 1 values).
00061  *
00062  */
00063 std::string hex_to_binary_string(char from);
00064
00065 /*!
00066  * \brief This function resamples a sequence of float values.
00067  *
00068  */
00069 void resampler(const own::span<float> from, own::span<float> dest,
00070     float fs_in, float fs_out);
00071
00072 /*!
00073  * \brief This function resamples a sequence of complex values.
00074  *
00075  */
00076 void resampler(own::span<const std::complex<float» from, own::span<std::complex<float» dest,
00077     float fs_in, float fs_out);
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_GNSS_SIGNAL_REPLICA_H
```

13.139 gnss_time.h File Reference

class that stores both the receiver time, relative to the receiver start and the GNSS time (absolute)

`#include <cstdint>`

Classes

- class GnssTime

13.139.1 Detailed Description

class that stores both the receiver time, relative to the receiver start and the GNSS time (absolute)

Author

Javier Arribas 2022. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_time.h.

13.140 gnss_time.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_time.h
00003  * \brief class that stores both the receiver time, relative to the receiver start and the GNSS time
      (absolute)
00004  * \author Javier Arribas 2022. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_TIME_H
00019 #define GNSS_SDR_GNSS_TIME_H
00020
00021 #include <cstdint>
00022
00023 class GnssTime
00024 {
00025 public:
00026     double rx_time;
00027     int week;               /*!< GPS week number (since January 1980) */
00028     int tow_ms;             /* time of week [ms]*/
00029     double tow_ms_fraction; /* tow ms fractional part [ms]*/
00030 };
00031
00032 #endif
```

13.141 gps_l2c_signal_replica.h File Reference

This file implements signal generators for GPS L2C signals.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void gps_l2c_m_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L2C M code for the desired SV ID.
- void gps_l2c_m_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates float GPS L2C M code for the desired SV ID.
- void gps_l2c_m_code_gen_float_cl_zeroed (own::span< float > dest, uint32_t prn)

  Generates float GPS L2C M code with double chip rate and L component zeroed for the desired SV ID.
- void gps_l2c_m_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency.
- void gps_l2c_m_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, bool cl_zeroed)

  Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency, optionally zeroing CL slots.

13.141.1 Detailed Description

This file implements signal generators for GPS L2C signals.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_l2c_signal_replica.h.

13.142 gps_l2c_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l2c_signal_replica.h
00003  * \brief This file implements signal generators for GPS L2C signals
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GPS_L2C_SIGNAL_REPLICA_H
00019 #define GNSS_SDR_GPS_L2C_SIGNAL_REPLICA_H
00020
00021 #include <complex>
00022 #include <cstdint>
00023 #if HAS_STD_SPAN
00024 #include <span>
00025 namespace own = std;
00026 #else
00027 #include <gsl-lite/gsl-lite.hpp>
00028 namespace own = gsl_lite;
00029 #endif
00030
00031 /** \addtogroup Algorithms_Library
00032  * \{ */
00033 /** \addtogroup Algorithm_libs algorithms_libs
00034  * \{ */
00035
00036
00037 //! Generates complex GPS L2C M code for the desired SV ID
00038 void gps_l2c_m_code_gen_complex(own::span<std::complex<float>> dest, uint32_t prn);
00039
00040 //! Generates float GPS L2C M code for the desired SV ID
00041 void gps_l2c_m_code_gen_float(own::span<float> dest, uint32_t prn);
00042
00043 //! Generates float GPS L2C M code with double chip rate and L component zeroed for the desired SV ID
00044 void gps_l2c_m_code_gen_float_cl_zeroed(own::span<float> dest, uint32_t prn);
00045
00046
00047 //! Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling frequency
00048 void gps_l2c_m_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq);
00049
00050 //! Generates complex GPS L2C M code for the desired SV ID, and sampled to specific sampling
      frequency, optionally zeroing CL slots
00051 void gps_l2c_m_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq, bool cl_zeroed);
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_GPS_L2C_SIGNAL_REPLICA_H
```

13.143 gps_l5_signal_replica.h File Reference

This file implements signal generators for GPS L5 signals.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void gps_l5i_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L5I code for the desired SV ID.
- void gps_l5i_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real GPS L5I code for the desired SV ID.
- void gps_l5q_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn)

  Generates complex GPS L5Q code for the desired SV ID.
- void gps_l5q_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real GPS L5Q code for the desired SV ID.
- void gps_l5i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L5I code for the desired SV ID, and sampled to specific sampling frequency.
- void gps_l5q_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex GPS L5Q code for the desired SV ID, and sampled to specific sampling frequency.

13.143.1 Detailed Description

This file implements signal generators for GPS L5 signals.

Author

Javier Arribas, 2017. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_l5_signal_replica.h.

13.144 gps_l5_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l5_signal_replica.h
00003  * \brief This file implements signal generators for GPS L5 signals
00004  * \author Javier Arribas, 2017. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GPS_L5_SIGNAL_REPLICA_H
00019 #define GNSS_SDR_GPS_L5_SIGNAL_REPLICA_H
00020
00021 #include <complex>
00022 #include <cstdint>
00023 #if HAS_STD_SPAN
00024 #include <span>
00025 namespace own = std;
00026 #else
00027 #include <gsl-lite/gsl-lite.hpp>
00028 namespace own = gsl_lite;
00029 #endif
00030
00031 /** \addtogroup Algorithms_Library
00032  * \{ */
00033 /** \addtogroup Algorithm_libs algorithms_libs
00034  * \{ */
00035
00036
00037 //! Generates complex GPS L5I code for the desired SV ID
00038 void gps_l5i_code_gen_complex(own::span<std::complex<float>> dest, uint32_t prn);
```

```
00039
00040 //! Generates real GPS L5I code for the desired SV ID
00041 void gps_l5i_code_gen_float(own::span<float> dest, uint32_t prn);
00042
00043 //! Generates complex GPS L5Q code for the desired SV ID
00044 void gps_l5q_code_gen_complex(own::span<std::complex<float» dest, uint32_t prn);
00045
00046 //! Generates real GPS L5Q code for the desired SV ID
00047 void gps_l5q_code_gen_float(own::span<float> dest, uint32_t prn);
00048
00049 //! Generates complex GPS L5I code for the desired SV ID, and sampled to specific sampling frequency
00050 void gps_l5i_code_gen_complex_sampled(own::span<std::complex<float» dest, uint32_t prn, int32_t
      sampling_freq);
00051
00052 //! Generates complex GPS L5Q code for the desired SV ID, and sampled to specific sampling frequency
00053 void gps_l5q_code_gen_complex_sampled(own::span<std::complex<float» dest, uint32_t prn, int32_t
      sampling_freq);
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_GPS_L5_SIGNAL_REPLICA_H
```

13.145 gps_sdr_signal_replica.h File Reference

This file implements functions for GPS L1 C/A signal replica generation.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void gps_l1_ca_code_gen_int (own::span< int32_t > dest, int32_t prn, uint32_t chip_shift)

  Generates int GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_float (own::span< float > dest, int32_t prn, uint32_t chip_shift)

  Generates float GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_complex (own::span< std::complex< float > > dest, int32_t prn, uint32_t chip↩_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift.
- void gps_l1_ca_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq, uint32_t chip_shift)

  Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific sampling frequency.

13.145.1 Detailed Description

This file implements functions for GPS L1 C/A signal replica generation.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_sdr_signal_replica.h.

13.146 gps_sdr_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_sdr_signal_replica.h
00003  * \brief This file implements functions for GPS L1 C/A signal replica
00004  * generation
00005  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00006  *
```

```
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GPS_SDR_SIGNAL_REPLICA_H
00020 #define GNSS_SDR_GPS_SDR_SIGNAL_REPLICA_H
00021
00022 #include <complex>
00023 #include <cstdint>
00024 #if HAS_STD_SPAN
00025 #include <span>
00026 namespace own = std;
00027 #else
00028 #include <gsl-lite/gsl-lite.hpp>
00029 namespace own = gsl_lite;
00030 #endif
00031
00032 /** \addtogroup Algorithms_Library
00033  * \{ */
00034 /** \addtogroup Algorithm_libs algorithms_libs
00035  * \{ */
00036
00037
00038 //! Generates int GPS L1 C/A code for the desired SV ID and code shift
00039 void gps_l1_ca_code_gen_int(own::span<int32_t> dest, int32_t prn, uint32_t chip_shift);
00040
00041 //! Generates float GPS L1 C/A code for the desired SV ID and code shift
00042 void gps_l1_ca_code_gen_float(own::span<float> dest, int32_t prn, uint32_t chip_shift);
00043
00044 //! Generates complex GPS L1 C/A code for the desired SV ID and code shift
00045 void gps_l1_ca_code_gen_complex(own::span<std::complex<float>> dest, int32_t prn, uint32_t chip_shift);
00046
00047 //! Generates complex GPS L1 C/A code for the desired SV ID and code shift, and sampled to specific
      sampling frequency
00048 void gps_l1_ca_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq, uint32_t chip_shift);
00049
00050
00051 /** \} */
00052 /** \} */
00053 #endif  // GNSS_SDR_GPS_SDR_SIGNAL_REPLICA_H
```

13.147 item_type_helpers.h File Reference

Utility functions for converting between item types.

```
#include <cstdint>
#include <functional>
#include <string>
```

Typedefs

- using item_type_converter_t = std::function<void(void *, const void *, uint32_t)>

Functions

- bool item_type_valid (const std::string &item_type)

  Check if a string is a valid item type.
- size_t item_type_size (const std::string &item_type)

  Return the size of the given item type, or zero if unknown.
- bool item_type_is_complex (const std::string &item_type)

  Determine if an item_type is complex.
- item_type_converter_t make_vector_converter (const std::string &input_type, const std::string &output_type)

  Create a function to convert an array of input_type to an array of output_type.

13.147.1 Detailed Description

Utility functions for converting between item types.

Authors

- Cillian O'Driscoll, 2019. cillian.odriscoll(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file item_type_helpers.h.

13.148 item_type_helpers.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file item_type_helpers.h
00003  * \brief Utility functions for converting between item types
00004  * \authors <ul>
00005  *          <li> Cillian O'Driscoll, 2019. cillian.odriscoll(at)gmail.com
00006  *          </ul>
00007  *
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_ITEM_TYPE_HELPERS_H
00021 #define GNSS_SDR_ITEM_TYPE_HELPERS_H
00022
00023
00024 #include <cstdint>
00025 #include <functional>
00026 #include <string>
00027
00028 /** \addtogroup Algorithms_Library
00029  * \{ */
00030 /** \addtogroup Algorithm_libs algorithms_libs
00031  * \{ */
00032
00033
00034 using item_type_converter_t = std::function<void(void *, const void *, uint32_t)>;
00035
00036 /*!
00037  * \brief Check if a string is a valid item type
00038  *
00039  * \description Valid item types include:
00040  *     "byte", "short", "float", "ibyte", "ishort", "cbyte", "cshort", "gr_complex"
00041  *
00042  */
00043 bool item_type_valid(const std::string &item_type);
00044
00045 /*!
00046  * \brief Return the size of the given item type, or zero if unknown
00047  */
00048 size_t item_type_size(const std::string &item_type);
00049
00050 /*!
00051  * \brief Determine if an item_type is complex
00052  */
00053 bool item_type_is_complex(const std::string &item_type);
00054
00055 /*!
00056  * \brief Create a function to convert an array of input_type to an array of output_type
00057  *
00058  * \description Provides a generic interface to generate conversion functions for mapping
00059  * arrays of items.
00060  *
00061  * \param input_type - String representation of the input item type
00062  * \param output_type - String representation of the output item type
00063  *
00064  * The item types accepted are:
00065  *
00066  *  1. "byte" for 8 bit integers
00067  *  2. "cbyte" for complex (interleaved) 8 bit integers
```

```
00068  *  4. "ibyte" for complex (interleaved) 8 bit integers
00069  *  4. "short" for 16 bit integers
00070  *  5. "cshort" for complex (interleaved) 16 bit integers
00071  *  6. "ishort" for complex (interleaved) 16 bit integers
00072  *  7. "float" for 32 bit floating point values
00073  *  8. "gr_complex" for complex (interleaved) 32 bit floating point values
00074  *
00075  * \returns A function object with the following prototype:
00076  *  void convert_fun( void *dest, void *src, int num_items );
00077  *
00078  */
00079 item_type_converter_t make_vector_converter(const std::string &input_type,
00080     const std::string &output_type);
00081
00082
00083 /** \} */
00084 /** \} */
00085 #endif  // GNSS_SDR_ITEM_TYPE_HELPERS_H
```

13.149 matlab_writter_helper.h File Reference

Utility functions for logging to a matlab file.

```
#include <matio.h>
#include <array>
```

Classes

- struct matlab_type_traits< int32_t >
- struct matlab_type_traits< uint32_t >
- struct matlab_type_traits< uint8_t >
- struct matlab_type_traits< int64_t >
- struct matlab_type_traits< uint64_t >
- struct matlab_type_traits< float >
- struct matlab_type_traits< double >

Functions

- template<size_t Rank, typename T>
  void write_matlab_var (const char *name, T data, mat_t *matfp, std::array< size_t, 2 > &dims)
- template<size_t Rank, typename T>
  void write_matlab_var (const char *name, T *data, mat_t *matfp, std::array< size_t, 2 > &dims)

13.149.1 Detailed Description

Utility functions for logging to a matlab file.

Authors

- Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file matlab_writter_helper.h.

13.149.2 Function Documentation

13.149.2.1 write_matlab_var() [1/2]

```
template<size_t Rank, typename T>
void write_matlab_var (
            const char * name,
            T * data,
            mat_t * matfp,
            std::array< size_t, 2 > & dims)
```

Definition at line 89 of file matlab_writter_helper.h.

13.149.2.2 write_matlab_var() [2/2]

```
template<size_t Rank, typename T>
void write_matlab_var (
             const char * name,
             T data,
             mat_t * matfp,
             std::array< size_t, 2 > & dims)
```

Definition at line 79 of file matlab_writter_helper.h.

13.150 matlab_writter_helper.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file matlab_writter_helper.h
00003  * \brief Utility functions for logging to a matlab file
00004  * \authors <ul>
00005  *          <li> Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com
00006  *          </ul>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_MATLAB_WRITTER_HELPER_H
00020 #define GNSS_SDR_MATLAB_WRITTER_HELPER_H
00021
00022 #include <matio.h>
00023 #include <array>
00024
00025 template <typename T>
00026 struct matlab_type_traits;
00027
00028 template <>
00029 struct matlab_type_traits<int32_t>
00030 {
00031     static constexpr matio_classes class_type = MAT_C_INT32;
00032     static constexpr matio_types data_type = MAT_T_INT32;
00033 };
00034
00035 template <>
00036 struct matlab_type_traits<uint32_t>
00037 {
00038     static constexpr matio_classes class_type = MAT_C_UINT32;
00039     static constexpr matio_types data_type = MAT_T_UINT32;
00040 };
00041
00042 template <>
00043 struct matlab_type_traits<uint8_t>
00044 {
00045     static constexpr matio_classes class_type = MAT_C_UINT8;
00046     static constexpr matio_types data_type = MAT_T_UINT8;
00047 };
00048
00049 template <>
00050 struct matlab_type_traits<int64_t>
00051 {
00052     static constexpr matio_classes class_type = MAT_C_INT64;
00053     static constexpr matio_types data_type = MAT_T_INT64;
00054 };
00055
00056 template <>
00057 struct matlab_type_traits<uint64_t>
00058 {
00059     static constexpr matio_classes class_type = MAT_C_UINT64;
00060     static constexpr matio_types data_type = MAT_T_UINT64;
00061 };
00062
00063 template <>
00064 struct matlab_type_traits<float>
00065 {
00066     static constexpr matio_classes class_type = MAT_C_SINGLE;
00067     static constexpr matio_types data_type = MAT_T_SINGLE;
```

```
00068 };
00069
00070 template <>
00071 struct matlab_type_traits<double>
00072 {
00073     static constexpr matio_classes class_type = MAT_C_DOUBLE;
00074     static constexpr matio_types data_type = MAT_T_DOUBLE;
00075 };
00076
00077
00078 template <size_t Rank, typename T>
00079 void write_matlab_var(const char* name, T data, mat_t* matfp, std::array<size_t, 2>& dims)
00080 {
00081     using traits = matlab_type_traits<T>;
00082     matvar_t* matvar = Mat_VarCreate(name, traits::class_type, traits::data_type, Rank, dims.data(),
      &data, 0);
00083     Mat_VarWrite(matfp, matvar, MAT_COMPRESSION_ZLIB);
00084     Mat_VarFree(matvar);
00085 }
00086
00087
00088 template <size_t Rank, typename T>
00089 void write_matlab_var(const char* name, T* data, mat_t* matfp, std::array<size_t, 2>& dims)
00090 {
00091     using traits = matlab_type_traits<T>;
00092     matvar_t* matvar = Mat_VarCreate(name, traits::class_type, traits::data_type, Rank, dims.data(),
      data, 0);
00093     Mat_VarWrite(matfp, matvar, MAT_COMPRESSION_ZLIB);
00094     Mat_VarFree(matvar);
00095 }
00096
00097 #endif  // GNSS_SDR_NAV_MESSAGE_MONITOR_H
```

13.151 clFFT.h File Reference

FFT in OpenCL.

```
#include <stdio.h>
#include <CL/cl.h>
```

Classes

- struct clFFT_Dim3
- struct clFFT_SplitComplex
- struct clFFT_Complex

Typedefs

- typedef void * clFFT_Plan

Enumerations

- enum **clFFT_Direction** { **clFFT_Forward** = -1 , **clFFT_Inverse** = 1 }
- enum **clFFT_Dimension** { **clFFT_1D** = 0 , **clFFT_2D** = 1 , **clFFT_3D** = 3 }
- enum **clFFT_DataFormat** { **clFFT_SplitComplexFormat** = 0 , **clFFT_InterleavedComplexFormat** = 1 }

Functions

- clFFT_Plan **clFFT_CreatePlan** (cl_context context, clFFT_Dim3 n, clFFT_Dimension dim, clFFT_Data↩ Format dataFormat, cl_int *error_code)
- void **clFFT_DestroyPlan** (clFFT_Plan plan)
- cl_int **clFFT_ExecuteInterleaved** (cl_command_queue queue, clFFT_Plan plan, cl_int batchSize, clFFT_↩ Direction dir, cl_mem data_in, cl_mem data_out, cl_int num_events, cl_event *event_list, cl_event *event)
- cl_int **clFFT_ExecutePlannar** (cl_command_queue queue, clFFT_Plan plan, cl_int batchSize, clFFT_↩ Direction dir, cl_mem data_in_real, cl_mem data_in_imag, cl_mem data_out_real, cl_mem data_out_imag, cl_int num_events, cl_event *event_list, cl_event *event)
- cl_int **clFFT_1DTwistInterleaved** (clFFT_Plan Plan, cl_command_queue queue, cl_mem array, size_t num↩ Rows, size_t numCols, size_t startRow, size_t rowsToProcess, clFFT_Direction dir)

- cl_int **clFFT_1DTwistPlannar** (clFFT_Plan Plan, cl_command_queue queue, cl_mem array_real, cl_mem array_imag, size_t numRows, size_t numCols, size_t startRow, size_t rowsToProcess, clFFT_Direction dir)
- void **clFFT_DumpPlan** (clFFT_Plan plan, FILE *file)

13.151.1 Detailed Description

FFT in OpenCL.
Version: <1.0>
Copyright (C) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive
Definition in file clFFT.h.

13.151.2 Typedef Documentation

13.151.2.1 clFFT_Plan

```
typedef void* clFFT_Plan
```
Definition at line 71 of file clFFT.h.

13.151.3 Enumeration Type Documentation

13.151.3.1 clFFT_DataFormat

```
enum clFFT_DataFormat
```
Definition at line 46 of file clFFT.h.

13.151.3.2 clFFT_Dimension

```
enum clFFT_Dimension
```
Definition at line 38 of file clFFT.h.

13.151.3.3 clFFT_Direction

```
enum clFFT_Direction
```
Definition at line 31 of file clFFT.h.

13.152 clFFT.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file clFFT.h
00003  * \brief FFT in OpenCL
00004  *
00005  *
00006  * Version:    <1.0>
00007  *
00008  * Copyright ( C ) 2008 Apple Inc. All Rights Reserved.
00009  * SPDX-License-Identifier: LicenseRef-Apple-Permissive
00010  *
00011  */
00012
00013 #ifndef __CLFFT_H
00014 #define __CLFFT_H
00015
00016 #ifdef __cplusplus
00017 extern "C"
00018 {
00019 #endif
00020
00021 #include <stdio.h>
00022
00023 #ifdef __APPLE__
00024 #define CL_SILENCE_DEPRECATION
00025 #include <OpenCL/opencl.h>
00026 #else
00027 #include <CL/cl.h>
00028 #endif
00029
00030     // XForm type
00031     typedef enum
```

```
00032     {
00033         clFFT_Forward = -1,
00034         clFFT_Inverse = 1
00035     } clFFT_Direction;
00036 
00037     // XForm dimension
00038     typedef enum
00039     {
00040         clFFT_1D = 0,
00041         clFFT_2D = 1,
00042         clFFT_3D = 3
00043     } clFFT_Dimension;
00044 
00045     // XForm Data type
00046     typedef enum
00047     {
00048         clFFT_SplitComplexFormat = 0,
00049         clFFT_InterleavedComplexFormat = 1
00050     } clFFT_DataFormat;
00051 
00052     typedef struct
00053     {
00054         unsigned int x;
00055         unsigned int y;
00056         unsigned int z;
00057     } clFFT_Dim3;
00058 
00059     typedef struct
00060     {
00061         float *real;
00062         float *imag;
00063     } clFFT_SplitComplex;
00064 
00065     typedef struct
00066     {
00067         float real;
00068         float imag;
00069     } clFFT_Complex;
00070 
00071     typedef void *clFFT_Plan;
00072 
00073     clFFT_Plan clFFT_CreatePlan(cl_context context, clFFT_Dim3 n, clFFT_Dimension dim,
      clFFT_DataFormat dataFormat, cl_int *error_code);
00074 
00075     void clFFT_DestroyPlan(clFFT_Plan plan);
00076 
00077     cl_int clFFT_ExecuteInterleaved(cl_command_queue queue, clFFT_Plan plan, cl_int batchSize,
      clFFT_Direction dir,
00078         cl_mem data_in, cl_mem data_out,
00079         cl_int num_events, cl_event *event_list, cl_event *event);
00080 
00081     cl_int clFFT_ExecutePlannar(cl_command_queue queue, clFFT_Plan plan, cl_int batchSize,
      clFFT_Direction dir,
00082         cl_mem data_in_real, cl_mem data_in_imag, cl_mem data_out_real, cl_mem data_out_imag,
00083         cl_int num_events, cl_event *event_list, cl_event *event);
00084 
00085     cl_int clFFT_1DTwistInterleaved(clFFT_Plan Plan, cl_command_queue queue, cl_mem array,
00086         size_t numRows, size_t numCols, size_t startRow, size_t rowsToProcess, clFFT_Direction dir);
00087 
00088 
00089     cl_int clFFT_1DTwistPlannar(clFFT_Plan Plan, cl_command_queue queue, cl_mem array_real, cl_mem
      array_imag,
00090         size_t numRows, size_t numCols, size_t startRow, size_t rowsToProcess, clFFT_Direction dir);
00091 
00092     void clFFT_DumpPlan(clFFT_Plan plan, FILE *file);
00093 
00094 #ifdef __cplusplus
00095 }
00096 #endif
00097 
00098 #endif
```

13.153 fft_base_kernels.h File Reference

FFT base kernels for OpenCL.

`#include <string>`

13.153.1 Detailed Description

FFT base kernels for OpenCL.

Version: <1.0>
Copyright (C) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive
Definition in file fft_base_kernels.h.

13.154 fft_base_kernels.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fft_base_kernels.h
00003  * \brief FFT base kernels for OpenCL
00004  *
00005  *
00006  * Version:    <1.0>
00007  *
00008  * Copyright ( C ) 2008 Apple Inc. All Rights Reserved.
00009  * SPDX-License-Identifier: LicenseRef-Apple-Permissive
00010  *
00011  */
00012
00013
00014 #ifndef __CL_FFT_BASE_KERNELS_
00015 #define __CL_FFT_BASE_KERNELS_
00016
00017 #include <string>
00018
00019 using namespace std;
00020
00021 static string baseKernels = string(
00022     "#ifndef M_PI\n"
00023     "#define M_PI 0x1.921fb54442d18p+1\n"
00024     "#endif\n"
00025     "#define complexMul(a,b) ((float2)(mad(-(a).y, (b).y, (a).x * (b).x), mad((a).y, (b).x, (a).x *
      (b).y)))\n"
00026     "#define conj(a) ((float2)((a).x, -(a).y))\n"
00027     "#define conjTransp(a) ((float2)(-(a).y, (a).x))\n"
00028     "\n"
00029     "#define fftKernel2(a,dir) \\\n"
00030     "{ \\\n"
00031     "    float2 c = (a)[0];    \\\n"
00032     "    (a)[0] = c + (a)[1];  \\\n"
00033     "    (a)[1] = c - (a)[1];  \\\n"
00034     "}\n"
00035     "\n"
00036     "#define fftKernel2S(d1,d2,dir) \\\n"
00037     "{ \\\n"
00038     "    float2 c = (d1);   \\\n"
00039     "    (d1) = c + (d2);   \\\n"
00040     "    (d2) = c - (d2);   \\\n"
00041     "}\n"
00042     "\n"
00043     "#define fftKernel4(a,dir) \\\n"
00044     "{ \\\n"
00045     "    fftKernel2S((a)[0], (a)[2], dir); \\\n"
00046     "    fftKernel2S((a)[1], (a)[3], dir); \\\n"
00047     "    fftKernel2S((a)[0], (a)[1], dir); \\\n"
00048     "    (a)[3] = (float2)(dir)*(conjTransp((a)[3])); \\\n"
00049     "    fftKernel2S((a)[2], (a)[3], dir); \\\n"
00050     "    float2 c = (a)[1]; \\\n"
00051     "    (a)[1] = (a)[2]; \\\n"
00052     "    (a)[2] = c; \\\n"
00053     "}\n"
00054     "\n"
00055     "#define fftKernel4s(a0,a1,a2,a3,dir) \\\n"
00056     "{ \\\n"
00057     "    fftKernel2S((a0), (a2), dir); \\\n"
00058     "    fftKernel2S((a1), (a3), dir); \\\n"
00059     "    fftKernel2S((a0), (a1), dir); \\\n"
00060     "    (a3) = (float2)(dir)*(conjTransp((a3))); \\\n"
00061     "    fftKernel2S((a2), (a3), dir); \\\n"
00062     "    float2 c = (a1); \\\n"
00063     "    (a1) = (a2); \\\n"
00064     "    (a2) = c; \\\n"
00065     "}\n"
00066     "\n"
00067     "#define bitreverse8(a) \\\n"
00068     "{ \\\n"
00069     "    float2 c; \\\n"
00070     "    c = (a)[1]; \\\n"
00071     "    (a)[1] = (a)[4]; \\\n"
00072     "    (a)[4] = c; \\\n"
00073     "    c = (a)[3]; \\\n"
00074     "    (a)[3] = (a)[6]; \\\n"
00075     "    (a)[6] = c; \\\n"
```

```
00076     "}\n"
00077     "\n"
00078     "#define fftKernel8(a,dir) \\\n"
00079     "{ \\\n"
00080     "   const float2 w1  = (float2)(0x1.6a09e6p-1f,  dir*0x1.6a09e6p-1f);  \\\n"
00081     "   const float2 w3  = (float2)(-0x1.6a09e6p-1f, dir*0x1.6a09e6p-1f);  \\\n"
00082     "   float2 c; \\\n"
00083     "   fftKernel2S((a)[0], (a)[4], dir); \\\n"
00084     "   fftKernel2S((a)[1], (a)[5], dir); \\\n"
00085     "   fftKernel2S((a)[2], (a)[6], dir); \\\n"
00086     "   fftKernel2S((a)[3], (a)[7], dir); \\\n"
00087     "   (a)[5] = complexMul(w1, (a)[5]); \\\n"
00088     "   (a)[6] = (float2)(dir)*(conjTransp((a)[6])); \\\n"
00089     "   (a)[7] = complexMul(w3, (a)[7]); \\\n"
00090     "   fftKernel2S((a)[0], (a)[2], dir); \\\n"
00091     "   fftKernel2S((a)[1], (a)[3], dir); \\\n"
00092     "   fftKernel2S((a)[4], (a)[6], dir); \\\n"
00093     "   fftKernel2S((a)[5], (a)[7], dir); \\\n"
00094     "   (a)[3] = (float2)(dir)*(conjTransp((a)[3])); \\\n"
00095     "   (a)[7] = (float2)(dir)*(conjTransp((a)[7])); \\\n"
00096     "   fftKernel2S((a)[0], (a)[1], dir); \\\n"
00097     "   fftKernel2S((a)[2], (a)[3], dir); \\\n"
00098     "   fftKernel2S((a)[4], (a)[5], dir); \\\n"
00099     "   fftKernel2S((a)[6], (a)[7], dir); \\\n"
00100     "   bitreverse8((a)); \\\n"
00101     "}\n"
00102     "\n"
00103     "#define bitreverse4x4(a) \\\n"
00104     "{ \\\n"
00105     "   float2 c; \\\n"
00106     "   c = (a)[1];  (a)[1]  = (a)[4];  (a)[4]  = c; \\\n"
00107     "   c = (a)[2];  (a)[2]  = (a)[8];  (a)[8]  = c; \\\n"
00108     "   c = (a)[3];  (a)[3]  = (a)[12]; (a)[12] = c; \\\n"
00109     "   c = (a)[6];  (a)[6]  = (a)[9];  (a)[9]  = c; \\\n"
00110     "   c = (a)[7];  (a)[7]  = (a)[13]; (a)[13] = c; \\\n"
00111     "   c = (a)[11]; (a)[11] = (a)[14]; (a)[14] = c; \\\n"
00112     "}\n"
00113     "\n"
00114     "#define fftKernel16(a,dir) \\\n"
00115     "{ \\\n"
00116     "    const float w0 = 0x1.d906bcp-1f; \\\n"
00117     "    const float w1 = 0x1.87de2ap-2f; \\\n"
00118     "    const float w2 = 0x1.6a09e6p-1f; \\\n"
00119     "    fftKernel4s((a)[0], (a)[4], (a)[8],  (a)[12], dir); \\\n"
00120     "    fftKernel4s((a)[1], (a)[5], (a)[9],  (a)[13], dir); \\\n"
00121     "    fftKernel4s((a)[2], (a)[6], (a)[10], (a)[14], dir); \\\n"
00122     "    fftKernel4s((a)[3], (a)[7], (a)[11], (a)[15], dir); \\\n"
00123     "    (a)[5]  = complexMul((a)[5], (float2)(w0, dir*w1)); \\\n"
00124     "    (a)[6]  = complexMul((a)[6], (float2)(w2, dir*w2)); \\\n"
00125     "    (a)[7]  = complexMul((a)[7], (float2)(w1, dir*w0)); \\\n"
00126     "    (a)[9]  = complexMul((a)[9], (float2)(w2, dir*w2)); \\\n"
00127     "    (a)[10] = (float2)(dir)*(conjTransp((a)[10])); \\\n"
00128     "    (a)[11] = complexMul((a)[11], (float2)(-w2, dir*w2)); \\\n"
00129     "    (a)[13] = complexMul((a)[13], (float2)(w1, dir*w0)); \\\n"
00130     "    (a)[14] = complexMul((a)[14], (float2)(-w2, dir*w2)); \\\n"
00131     "    (a)[15] = complexMul((a)[15], (float2)(-w0, dir*-w1)); \\\n"
00132     "    fftKernel4((a), dir); \\\n"
00133     "    fftKernel4((a) + 4, dir); \\\n"
00134     "    fftKernel4((a) + 8, dir); \\\n"
00135     "    fftKernel4((a) + 12, dir); \\\n"
00136     "    bitreverse4x4((a)); \\\n"
00137     "}\n"
00138     "\n"
00139     "#define bitreverse32(a) \\\n"
00140     "{ \\\n"
00141     "    float2 c1, c2; \\\n"
00142     "    c1 = (a)[2];   (a)[2] = (a)[1];   c2 = (a)[4];   (a)[4] = c1;   c1 = (a)[8];   (a)[8] = c2;
      c2 = (a)[16];  (a)[16] = c1;   (a)[1] = c2; \\\n"
00143     "    c1 = (a)[6];   (a)[6] = (a)[3];   c2 = (a)[12];  (a)[12] = c1;  c1 = (a)[24];  (a)[24] = c2;
      c2 = (a)[17];  (a)[17] = c1;   (a)[3] = c2; \\\n"
00144     "    c1 = (a)[10];  (a)[10] = (a)[5];  c2 = (a)[20];  (a)[20] = c1;  c1 = (a)[9];   (a)[9] = c2;
      c2 = (a)[18];  (a)[18] = c1;   (a)[5] = c2; \\\n"
00145     "    c1 = (a)[14];  (a)[14] = (a)[7];  c2 = (a)[28];  (a)[28] = c1;  c1 = (a)[25];  (a)[25] = c2;
      c2 = (a)[19];  (a)[19] = c1;   (a)[7] = c2; \\\n"
00146     "    c1 = (a)[22];  (a)[22] = (a)[11]; c2 = (a)[13];  (a)[13] = c1;  c1 = (a)[26];  (a)[26] = c2;
      c2 = (a)[21];  (a)[21] = c1;   (a)[11] = c2; \\\n"
00147     "    c1 = (a)[30];  (a)[30] = (a)[15]; c2 = (a)[29];  (a)[29] = c1;  c1 = (a)[27];  (a)[27] = c2;
      c2 = (a)[23];  (a)[23] = c1;   (a)[15] = c2; \\\n"
00148     "}\n"
00149     "\n"
00150     "#define fftKernel32(a,dir) \\\n"
00151     "{ \\\n"
00152     "    fftKernel2S((a)[0],  (a)[16], dir); \\\n"
00153     "    fftKernel2S((a)[1],  (a)[17], dir); \\\n"
00154     "    fftKernel2S((a)[2],  (a)[18], dir); \\\n"
00155     "    fftKernel2S((a)[3],  (a)[19], dir); \\\n"
00156     "    fftKernel2S((a)[4],  (a)[20], dir); \\\n"
```

```
00157     "    fftKernel2S((a)[5],  (a)[21], dir); \\\n"
00158     "    fftKernel2S((a)[6],  (a)[22], dir); \\\n"
00159     "    fftKernel2S((a)[7],  (a)[23], dir); \\\n"
00160     "    fftKernel2S((a)[8],  (a)[24], dir); \\\n"
00161     "    fftKernel2S((a)[9],  (a)[25], dir); \\\n"
00162     "    fftKernel2S((a)[10], (a)[26], dir); \\\n"
00163     "    fftKernel2S((a)[11], (a)[27], dir); \\\n"
00164     "    fftKernel2S((a)[12], (a)[28], dir); \\\n"
00165     "    fftKernel2S((a)[13], (a)[29], dir); \\\n"
00166     "    fftKernel2S((a)[14], (a)[30], dir); \\\n"
00167     "    fftKernel2S((a)[15], (a)[31], dir); \\\n"
00168     "    (a)[17] = complexMul((a)[17], (float2)(0x1.f6297cp-1f, dir*0x1.8f8b84p-3f)); \\\n"
00169     "    (a)[18] = complexMul((a)[18], (float2)(0x1.d906bcp-1f, dir*0x1.87de2ap-2f)); \\\n"
00170     "    (a)[19] = complexMul((a)[19], (float2)(0x1.a9b662p-1f, dir*0x1.1c73b4p-1f)); \\\n"
00171     "    (a)[20] = complexMul((a)[20], (float2)(0x1.6a09e6p-1f, dir*0x1.6a09e6p-1f)); \\\n"
00172     "    (a)[21] = complexMul((a)[21], (float2)(0x1.1c73b4p-1f, dir*0x1.a9b662p-1f)); \\\n"
00173     "    (a)[22] = complexMul((a)[22], (float2)(0x1.87de2ap-2f, dir*0x1.d906bcp-1f)); \\\n"
00174     "    (a)[23] = complexMul((a)[23], (float2)(0x1.8f8b84p-3f, dir*0x1.f6297cp-1f)); \\\n"
00175     "    (a)[24] = complexMul((a)[24], (float2)(0x0p+0f, dir*0x1p+0f)); \\\n"
00176     "    (a)[25] = complexMul((a)[25], (float2)(-0x1.8f8b84p-3f, dir*0x1.f6297cp-1f)); \\\n"
00177     "    (a)[26] = complexMul((a)[26], (float2)(-0x1.87de2ap-2f, dir*0x1.d906bcp-1f)); \\\n"
00178     "    (a)[27] = complexMul((a)[27], (float2)(-0x1.1c73b4p-1f, dir*0x1.a9b662p-1f)); \\\n"
00179     "    (a)[28] = complexMul((a)[28], (float2)(-0x1.6a09e6p-1f, dir*0x1.6a09e6p-1f)); \\\n"
00180     "    (a)[29] = complexMul((a)[29], (float2)(-0x1.a9b662p-1f, dir*0x1.1c73b4p-1f)); \\\n"
00181     "    (a)[30] = complexMul((a)[30], (float2)(-0x1.d906bcp-1f, dir*0x1.87de2ap-2f)); \\\n"
00182     "    (a)[31] = complexMul((a)[31], (float2)(-0x1.f6297cp-1f, dir*0x1.8f8b84p-3f)); \\\n"
00183     "    fftKernel16((a), dir); \\\n"
00184     "    fftKernel16((a) + 16, dir); \\\n"
00185     "    bitreverse32((a)); \\\n"
00186     "}\n\n");
00187
00188 static string twistKernelInterleaved = string(
00189     "__kernel void \\\n"
00190     "clFFT_1DTwistInterleaved(__global float2 *in, unsigned int startRow, unsigned int numCols,
      unsigned int N, unsigned int numRowsToProcess, int dir) \\\n"
00191     "{ \\\n"
00192     "   float2 a, w; \\\n"
00193     "   float ang; \\\n"
00194     "   unsigned int j; \\\n"
00195     "   unsigned int i = get_global_id(0); \\\n"
00196     "   unsigned int startIndex = i; \\\n"
00197     "    \\\n"
00198     "   if(i < numCols) \\\n"
00199     "   { \\\n"
00200     "       for(j = 0; j < numRowsToProcess; j++) \\\n"
00201     "       { \\\n"
00202     "           a = in[startIndex]; \\\n"
00203     "           ang = 2.0f * M_PI * dir * i * (startRow + j) / N; \\\n"
00204     "           w = (float2)(native_cos(ang), native_sin(ang)); \\\n"
00205     "           a = complexMul(a, w); \\\n"
00206     "           in[startIndex] = a; \\\n"
00207     "           startIndex += numCols; \\\n"
00208     "       } \\\n"
00209     "   }    \\\n"
00210     "} \\\n");
00211
00212 static string twistKernelPlannar = string(
00213     "__kernel void \\\n"
00214     "clFFT_1DTwistSplit(__global float *in_real, __global float *in_imag , unsigned int startRow,
      unsigned int numCols, unsigned int N, unsigned int numRowsToProcess, int dir) \\\n"
00215     "{ \\\n"
00216     "    float2 a, w; \\\n"
00217     "    float ang; \\\n"
00218     "    unsigned int j; \\\n"
00219     "   unsigned int i = get_global_id(0); \\\n"
00220     "   unsigned int startIndex = i; \\\n"
00221     "    \\\n"
00222     "   if(i < numCols) \\\n"
00223     "   { \\\n"
00224     "       for(j = 0; j < numRowsToProcess; j++) \\\n"
00225     "       { \\\n"
00226     "           a = (float2)(in_real[startIndex], in_imag[startIndex]); \\\n"
00227     "           ang = 2.0f * M_PI * dir * i * (startRow + j) / N; \\\n"
00228     "           w = (float2)(native_cos(ang), native_sin(ang)); \\\n"
00229     "           a = complexMul(a, w); \\\n"
00230     "           in_real[startIndex] = a.x; \\\n"
00231     "           in_imag[startIndex] = a.y; \\\n"
00232     "           startIndex += numCols; \\\n"
00233     "       } \\\n"
00234     "   }    \\\n"
00235     "} \\\n");
00236
00237
00238 #endif
```

13.155 fft_internal.h File Reference

Internals of FFT for OpenCL.

```
#include "clFFT.h"
#include <iostream>
#include <sstream>
#include <string>
```

Classes

- struct kernel_info_t
- struct cl_fft_plan

Typedefs

- typedef enum kernel_dir_t **cl_fft_kernel_dir**
- typedef struct kernel_info_t **cl_fft_kernel_info**

Enumerations

- enum **kernel_dir_t** { **cl_fft_kernel_x** , **cl_fft_kernel_y** , **cl_fft_kernel_z** }

Functions

- void **FFT1D** (cl_fft_plan ∗plan, cl_fft_kernel_dir dir)

13.155.1 Detailed Description

Internals of FFT for OpenCL.
Version: <1.0>
Copyright (C) 2008 Apple Inc. All Rights Reserved. SPDX-License-Identifier: LicenseRef-Apple-Permissive
Definition in file fft_internal.h.

13.155.2 Enumeration Type Documentation

13.155.2.1 kernel_dir_t

```
enum kernel_dir_t
```

Definition at line 25 of file fft_internal.h.

13.156 fft_internal.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fft_internal.h
00003  * \brief Internals of FFT for OpenCL
00004  *
00005  *
00006  * Version:    <1.0>
00007  *
00008  * Copyright ( C ) 2008 Apple Inc. All Rights Reserved.
00009  * SPDX-License-Identifier: LicenseRef-Apple-Permissive
00010  *
00011  *
00012  */
00013
00014
00015 #ifndef __CLFFT_INTERNAL_H
00016 #define __CLFFT_INTERNAL_H
00017
00018 #include "clFFT.h"
00019 #include <iostream>
00020 #include <sstream>
00021 #include <string>
```

```
00022
00023 using namespace std;
00024
00025 typedef enum kernel_dir_t
00026 {
00027     cl_fft_kernel_x,
00028     cl_fft_kernel_y,
00029     cl_fft_kernel_z
00030 } cl_fft_kernel_dir;
00031
00032 typedef struct kernel_info_t
00033 {
00034     cl_kernel kernel;
00035     char *kernel_name;
00036     unsigned lmem_size;
00037     unsigned num_workgroups;
00038     unsigned num_xforms_per_workgroup;
00039     unsigned num_workitems_per_workgroup;
00040     cl_fft_kernel_dir dir;
00041     int in_place_possible;
00042     kernel_info_t *next;
00043 } cl_fft_kernel_info;
00044
00045 typedef struct
00046 {
00047     // context in which fft resources are created and kernels are executed
00048     cl_context context;
00049
00050     // size of signal
00051     clFFT_Dim3 n;
00052
00053     // dimension of transform ... must be either 1D, 2D or 3D
00054     clFFT_Dimension dim;
00055
00056     // data format ... must be either interleaved or plannar
00057     clFFT_DataFormat format;
00058
00059     // string containing kernel source. Generated at runtime based on
00060     // n, dim, format and other parameters
00061     string *kernel_string;
00062
00063     // CL program containing source and kernel this particular
00064     // n, dim, data format
00065     cl_program program;
00066
00067     // linked list of kernels which needs to be executed for this fft
00068     cl_fft_kernel_info *kernel_info;
00069
00070     // number of kernels
00071     int num_kernels;
00072
00073     // twist kernel for virtualizing fft of very large sizes that do not
00074     // fit in GPU global memory
00075     cl_kernel twist_kernel;
00076
00077     // flag indicating if temporary intermediate buffer is needed or not.
00078     // this depends on fft kernels being executed and if transform is
00079     // in-place or out-of-place. e.g. Local memory fft (say 1D 1024 ...
00080     // one that does not require global transpose do not need temporary buffer)
00081     // 2D 1024x1024 out-of-place fft however do require intermediate buffer.
00082     // If temp buffer is needed, its allocation is lazy i.e. its not allocated
00083     // until its needed
00084     cl_int temp_buffer_needed;
00085
00086     // Batch size is runtime parameter and size of temporary buffer (if needed)
00087     // depends on batch size. Allocation of temporary buffer is lazy i.e. its
00088     // only created when needed. Once its created at first call of clFFT_Executexxx
00089     // it is not allocated next time if next time clFFT_Executexxx is called with
00090     // batch size different than the first call. last_batch_size caches the last
00091     // batch size with which this plan is used so that we dont keep allocating/deallocating
00092     // temp buffer if same batch size is used again and again.
00093     unsigned last_batch_size;
00094
00095     // temporary buffer for interleaved plan
00096     cl_mem tempmemobj;
00097
00098     // temporary buffer for planner plan. Only one of tempmemobj or
00099     // (tempmemobj_real, tempmemobj_imag) pair is valid (allocated) depending
00100     // data format of plan (plannar or interleaved)
00101     cl_mem tempmemobj_real, tempmemobj_imag;
00102
00103     // Maximum size of signal for which local memory transposed based
00104     // fft is sufficient i.e. no global mem transpose (communication)
00105     // is needed
00106     unsigned max_localmem_fft_size;
00107
00108     // Maximum work items per work group allowed. This, along with max_radix below controls
```

```
00109     // maximum local memory being used by fft kernels of this plan. Set to 256 by default
00110     unsigned max_work_item_per_workgroup;
00111
00112     // Maximum base radix for local memory fft ... this controls the maximum register
00113     // space used by work items. Currently defaults to 16
00114     unsigned max_radix;
00115
00116     // Device depended parameter that tells how many work-items need to be read consecutive
00117     // values to make sure global memory access by work-items of a work-group result in
00118     // coalesced memory access to utilize full bandwidth e.g. on NVidia tesla, this is 16
00119     unsigned min_mem_coalesce_width;
00120
00121     // Number of local memory banks. This is used to geneate kernel with local memory
00122     // transposes with appropriate padding to avoid bank conflicts to local memory
00123     // e.g. on NVidia it is 16.
00124     unsigned num_local_mem_banks;
00125 } cl_fft_plan;
00126
00127 void FFT1D(cl_fft_plan *plan, cl_fft_kernel_dir dir);
00128
00129 #endif
```

13.157 pass_through.h File Reference

Interface of a block that just puts its input in its output.

```
#include "conjugate_cc.h"
#include "conjugate_ic.h"
#include "conjugate_sc.h"
#include "gnss_block_interface.h"
#include <gnuradio/blocks/copy.h>
#include <gnuradio/runtime_types.h>
#include <cstddef>
#include <string>
```

Classes

- class Pass_Through

  This class implements a block that connects input and output (does nothing).

13.157.1 Detailed Description

Interface of a block that just puts its input in its output.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pass_through.h.

13.158 pass_through.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pass_through.h
00003  * \brief Interface of a block that just puts its input in its
00004  *        output.
00005  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
```

```
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_PASS_THROUGH_H
00020 #define GNSS_SDR_PASS_THROUGH_H
00021
00022 #include "conjugate_cc.h"
00023 #include "conjugate_ic.h"
00024 #include "conjugate_sc.h"
00025 #include "gnss_block_interface.h"
00026 #include <gnuradio/blocks/copy.h>
00027 #include <gnuradio/runtime_types.h>
00028 #include <cstddef>
00029 #include <string>
00030
00031 /** \addtogroup Algorithms_Library
00032  * \{ */
00033 /** \addtogroup Algorithm_libs algorithms_libs
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 /*!
00040  * \brief This class implements a block that connects input and output (does nothing)
00041  */
00042 class Pass_Through : public GNSSBlockInterface
00043 {
00044 public:
00045     Pass_Through(const ConfigurationInterface* configuration,
00046         const std::string& role,
00047         unsigned int in_stream,
00048         unsigned int out_stream);
00049
00050     ~Pass_Through() = default;
00051
00052     inline std::string role() override
00053     {
00054         return role_;
00055     }
00056
00057     //! Returns "Pass_Through"
00058     inline std::string implementation() override
00059     {
00060         return "Pass_Through";
00061     }
00062
00063     inline std::string item_type() const
00064     {
00065         return item_type_;
00066     }
00067
00068     inline size_t item_size() override
00069     {
00070         return item_size_;
00071     }
00072
00073     void connect(gr::top_block_sptr top_block) override;
00074     void disconnect(gr::top_block_sptr top_block) override;
00075     gr::basic_block_sptr get_left_block() override;
00076     gr::basic_block_sptr get_right_block() override;
00077
00078 private:
00079     gr::blocks::copy::sptr kludge_copy_;
00080     conjugate_cc_sptr conjugate_cc_;
00081     conjugate_sc_sptr conjugate_sc_;
00082     conjugate_ic_sptr conjugate_ic_;
00083     std::string item_type_;
00084     std::string role_;
00085     size_t item_size_;
00086     unsigned int in_streams_;
00087     unsigned int out_streams_;
00088     bool inverted_spectrum;
00089 };
00090
00091
00092 /** \} */
00093 /** \} */
00094 #endif  // GNSS_SDR_PASS_THROUGH_H
```

13.159 qzss_signal_replica.h File Reference

This file implements signal generators for QZSS signals.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- bool qzss_l1_code_is_boc (uint32_t prn)

  Returns true if the given QZSS PRN broadcasts the BOC(1,1)-modulated L1 C/B signal (IS-QZSS-PNT-006).
- void qzss_l1_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L1 C/A(B) code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l1_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L1 C/A(B) code for the desired SV ID (QZSS_L1_SAMPLES_PER_CHIP samples per chip).
- void qzss_l5i_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L5I code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l5i_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L5I code for the desired SV ID.
- void qzss_l5q_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex QZSS L5Q code for the desired SV ID, and sampled to specific sampling frequency.
- void qzss_l5q_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real QZSS L5I code for the desired SV ID.

13.159.1 Detailed Description

This file implements signal generators for QZSS signals.

Author

Carles Fernández-Prades, 2026. cfernandez (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file qzss_signal_replica.h.

13.160 qzss_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_signal_replica.h
00003  * \brief This file implements signal generators for QZSS signals
00004  * \author Carles Fernández-Prades, 2026. cfernandez (at) cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_QZSS_SIGNAL_REPLICA_H
00018 #define GNSS_SDR_QZSS_SIGNAL_REPLICA_H
00019
00020 #include <complex>
00021 #include <cstdint>
00022 #if HAS_STD_SPAN
00023 #include <span>
00024 namespace own = std;
00025 #else
00026 #include <gsl-lite/gsl-lite.hpp>
```

```
00027 namespace own = gsl_lite;
00028 #endif
00029
00030 /** \addtogroup Algorithms_Library
00031  * \{ */
00032 /** \addtogroup Algorithm_libs algorithms_libs
00033  * \{ */
00034
00035 //! Returns true if the given QZSS PRN broadcasts the BOC(1,1)-modulated L1 C/B signal
      (IS-QZSS-PNT-006)
00036 bool qzss_l1_code_is_boc(uint32_t prn);
00037
00038 //! Generates complex QZSS L1 C/A(B) code for the desired SV ID, and sampled to specific sampling
      frequency
00039 void qzss_l1_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq);
00040
00041 //! Generates real QZSS L1 C/A(B) code for the desired SV ID (QZSS_L1_SAMPLES_PER_CHIP samples per
      chip)
00042 void qzss_l1_code_gen_float(own::span<float> dest, uint32_t prn);
00043
00044 //! Generates complex QZSS L5I code for the desired SV ID, and sampled to specific sampling frequency
00045 void qzss_l5i_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq);
00046
00047 //! Generates real QZSS L5I code for the desired SV ID
00048 void qzss_l5i_code_gen_float(own::span<float> dest, uint32_t prn);
00049
00050 //! Generates complex QZSS L5Q code for the desired SV ID, and sampled to specific sampling frequency
00051 void qzss_l5q_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq);
00052
00053 //! Generates real QZSS L5I code for the desired SV ID
00054 void qzss_l5q_code_gen_float(own::span<float> dest, uint32_t prn);
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_QZSS_SIGNAL_REPLICA_H
```

13.161 rtklib.h File Reference

main header file for the rtklib library

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include "gnss_obs_codes.h"
#include <cctype>
#include <cmath>
#include <cstdarg>
#include <cstdint>
#include <cstdlib>
#include <netinet/in.h>
#include <pthread.h>
#include <string>
```

Classes

- struct gtime_t
- struct obsd_t
- struct obs_t
- struct erpd_t
- struct erp_t
- struct pcv_t
- struct pcvs_t
- struct alm_t
- struct eph_t
- struct geph_t
- struct peph_t
- struct pclk_t

- struct seph_t
- struct tled_t
- struct tle_t
- struct tec_t
- struct fcbd_t
- struct sbsmsg_t
- struct sbs_t
- struct sbsfcorr_t
- struct sbslcorr_t
- struct sbssatp_t
- struct sbssat_t
- struct sbsigp_t
- struct sbsigpband_t
- struct sbsion_t
- struct dgps_t
- struct ssr_t
- struct lexmsg_t
- struct lex_t
- struct lexeph_t
- struct lexion_t
- struct stec_t
- struct trop_t
- struct pppcorr_t
- struct nav_t
- struct sta_t
- struct sol_t
- struct solbuf_t
- struct solstat_t
- struct solstatbuf_t
- struct rtcm_t
- struct url_t
- struct opt_t
- struct exterr_t
- struct snrmask_t
- struct prcopt_t
- struct solopt_t
- struct ssat_t
- struct ambc_t
- struct rtk_t
- struct half_cyc_tag
- struct stream_t
- struct serial_t
- struct file_t
- struct tcp_t
- struct tcpsvr_t
- struct tcpcli_t
- struct ntrip_t
- struct ftp_t
- struct raw_t
- struct rtksvr_t
- struct msm_h_t

Macros

- #define dev_t int
- #define socket_t int
- #define closesocket close
- #define lock_t pthread_mutex_t
- #define initlock(f)
- #define rtk_lock(f)
- #define rtk_unlock(f)
- #define VER_RTKLIB "2.4.2"
- #define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
- #define NTRIP_CLI_PORT 2101
- #define NTRIP_SVR_PORT 80
- #define NTRIP_MAXRSP 32768
- #define NTRIP_MAXSTR 256
- #define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"
- #define NTRIP_RSP_OK_SVR "OK\r\n"
- #define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n"
- #define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
- #define NTRIP_RSP_HTTP "HTTP/"
- #define NTRIP_RSP_ERROR "ERROR"
- #define FTP_CMD "wget"
- #define ENAGLO
- #define ENAQZS
- #define ENABDS
- #define STR_MODE_R 0x1
- #define STR_MODE_W 0x2
- #define STR_MODE_RW 0x3
- #define STR_NONE 0
- #define STR_SERIAL 1
- #define STR_FILE 2
- #define STR_TCPSVR 3
- #define STR_TCPCLI 4
- #define STR_UDP 5
- #define STR_NTRIPSVR 6
- #define STR_NTRIPCLI 7
- #define STR_FTP 8
- #define STR_HTTP 9
- #define NP_PPP(opt)
- #define IC_PPP(s, opt)
- #define IT_PPP(opt)
- #define NR_PPP(opt)
- #define IB_PPP(s, opt)
- #define NX_PPP(opt)
- #define NF_RTK(opt)
- #define NP_RTK(opt)
- #define NI_RTK(opt)
- #define NT_RTK(opt)
- #define NL_RTK(opt)
- #define NB_RTK(opt)
- #define NR_RTK(opt)
- #define NX_RTK(opt)

Typedefs

- using fatalfunc_t = void(const char *)

  fatal callback function type
- typedef struct half_cyc_tag **half_cyc_t**

Functions

- double sysefact (int sys)

  Measurement error factor of a navigation system (single home of the sys-to-EFACT mapping used by the single-point, relative and PPP error models).

Variables

- const int TINTACT = 200

  period for stream active (ms)
- const int SERIBUFFSIZE = 4096

  serial buffer size (bytes)
- const int TIMETAGH_LEN = 64

  time tag file header length
- const int MAXCLI = 32

  max client connection for tcp svr
- const int MAXSTATMSG = 32

  max length of status message
- const int FTP_TIMEOUT = 30

  ftp/http timeout (s)
- const int MAXRAWLEN = 4096

  max length of receiver raw message
- const int MAXSOLBUF = 256

  max number of solution buffer
- const int MAXSBSMSG = 32

  max number of SBAS msg in RTK server
- const int MAXOBSBUF = 128

  max number of observation data buffer
- const int FILEPATHSEP = '/'
- const double RE_WGS84 = 6378137.0

  earth semimajor axis (WGS84) (m)
- const double FE_WGS84 = (1.0 / 298.257223563)

  earth flattening (WGS84)
- const double HION = 350000.0

  ionosphere height (m)
- const double PRN_HWBIAS = 1e-6

  process noise of h/w bias (m/MHz/sqrt(s))
- const double INT_SWAP_STAT = 86400.0

  swap interval of solution status file (s)
- const double INT_SWAP_TRAC = 86400.0

  swap interval of trace file (s)
- const unsigned int POLYCRC32 = 0xEDB88320u

  CRC32 polynomial.
- const unsigned int POLYCRC24Q = 0x1864CFBu

  CRC24Q polynomial.
- const int PMODE_SINGLE = 0

  positioning mode: single

- const int PMODE_DGPS = 1
  positioning mode: DGPS/DGNSS
- const int PMODE_KINEMA = 2
  positioning mode: kinematic
- const int PMODE_STATIC = 3
  positioning mode: static
- const int PMODE_MOVEB = 4
  positioning mode: moving-base
- const int PMODE_FIXED = 5
  positioning mode: fixed
- const int PMODE_PPP_KINEMA = 6
  positioning mode: PPP-kinemaric
- const int PMODE_PPP_STATIC = 7
  positioning mode: PPP-static
- const int PMODE_PPP_FIXED = 8
  positioning mode: PPP-fixed
- const int SOLF_LLH = 0
  solution format: lat/lon/height
- const int SOLF_XYZ = 1
  solution format: x/y/z-ecef
- const int SOLF_ENU = 2
  solution format: e/n/u-baseline
- const int SOLF_NMEA = 3
  solution format: NMEA-183
- const int SOLF_STAT = 4
  solution format: solution status
- const int SOLF_GSIF = 5
  solution format: GSI F1/F2
- const int SOLQ_NONE = 0
  solution status: no solution
- const int SOLQ_FIX = 1
  solution status: fix
- const int SOLQ_FLOAT = 2
  solution status: float
- const int SOLQ_SBAS = 3
  solution status: SBAS
- const int SOLQ_DGPS = 4
  solution status: DGPS/DGNSS
- const int SOLQ_SINGLE = 5
  solution status: single
- const int SOLQ_PPP = 6
  solution status: PPP
- const int SOLQ_DR = 7
  solution status: dead reckoning
- const int MAXSOLQ = 7
  max number of solution status
- const int TIMES_GPST = 0
  time system: gps time
- const int TIMES_UTC = 1
  time system: utc
- const int TIMES_JST = 2

time system: jst

- const double ERR_SAAS = 0.3

  saastamoinen model error std (m)
- const double ERR_BRDCI = 0.5

  broadcast iono model error factor
- const double ERR_CBIAS = 0.3

  code bias error std (m)
- const double REL_HUMI = 0.7

  relative humidity for saastamoinen model
- const double GAP_RESION = 120

  default gap to reset ionos parameters (ep)
- const int MAXFREQ = 7

  max NFREQ
- const int MAXLEAPS = 64

  max number of leap seconds table
- const double DTTOL = 0.005

  tolerance of time difference (s)
- const int NFREQ = 3

  number of carrier frequencies
- const int NFREQGLO = 2

  number of carrier frequencies of GLONASS
- const int NEXOBS = 0

  number of extended obs codes
- const int MAXANT = 64

  max length of station name/antenna type
- const int MINPRNGPS = 1

  min satellite PRN number of GPS
- const int MAXPRNGPS = 32

  max satellite PRN number of GPS
- const int NSATGPS = (MAXPRNGPS - MINPRNGPS + 1)

  number of GPS satellites
- const int NSYSGPS = 1
- const int SYS_NONE = 0x00

  navigation system: none
- const int SYS_GPS = 0x01

  navigation system: GPS
- const int SYS_SBS = 0x02

  navigation system: SBAS
- const int SYS_GLO = 0x04

  navigation system: GLONASS
- const int SYS_GAL = 0x08

  navigation system: Galileo
- const int SYS_QZS = 0x10

  navigation system: QZSS
- const int SYS_BDS = 0x20

  navigation system: BeiDou
- const int SYS_IRN = 0x40

  navigation system: IRNS
- const int SYS_LEO = 0x80

  navigation system: LEO
- const int SYS_ALL = 0xFF

navigation system: all

- const int MINPRNGLO = 1

  min satellite slot number of GLONASS
- const int MAXPRNGLO = 27

  max satellite slot number of GLONASS
- const int NSATGLO = (MAXPRNGLO - MINPRNGLO + 1)

  number of GLONASS satellites
- const int NSYSGLO = 1
- const int MINPRNGAL = 1

  min satellite PRN number of Galileo
- const int MAXPRNGAL = 36

  max satellite PRN number of Galileo
- const int NSATGAL = (MAXPRNGAL - MINPRNGAL + 1)

  number of Galileo satellites
- const int NSYSGAL = 1
- const int MAXPRNQZS = 206

  max satellite PRN number of QZSS
- const int MINPRNQZS = 193

  min satellite PRN number of QZSS
- const int MINPRNQZS_S = 183

  min satellite PRN number of QZSS SAIF
- const int MAXPRNQZS_S = 191

  max satellite PRN number of QZSS SAIF
- const int NSATQZS = (MAXPRNQZS - MINPRNQZS + 1)

  number of QZSS satellites
- const int NSYSQZS = 1
- const int MINPRNBDS = 1

  min satellite sat number of BeiDou
- const int MAXPRNBDS = 63

  max satellite sat number of BeiDou
- const int NSATBDS = (MAXPRNBDS - MINPRNBDS + 1)

  number of BeiDou satellites
- const int NSYSBDS = 1
- const int MINPRNIRN = 1

  min satellite sat number of IRNSS
- const int MAXPRNIRN = 7

  max satellite sat number of IRNSS
- const int NSATIRN = 0
- const int NSYSIRN = 0
- const int MINPRNLEO = 1

  min satellite sat number of LEO
- const int MAXPRNLEO = 10

  max satellite sat number of LEO ∗/
- const int NSATLEO = 0
- const int NSYSLEO = 0
- const int NSYS = (NSYSGPS + NSYSGLO + NSYSGAL + NSYSQZS + NSYSBDS + NSYSIRN + NSYSLEO)

  number of systems
- const int MINPRNSBS = 120

  min satellite PRN number of SBAS
- const int MAXPRNSBS = 142

  max satellite PRN number of SBAS

- const int NSATSBS = (MAXPRNSBS - MINPRNSBS + 1)

  number of SBAS satellites
- const int MAXSAT = (NSATGPS + NSATGLO + NSATGAL + NSATQZS + NSATBDS + NSATIRN + NSATSBS + NSATLEO)
- const int MAXSTA = 255
- const int MAXOBS = 64

  max number of obs in an epoch
- const int MAXRCV = 64

  max receiver number (1 to MAXRCV)
- const int MAXOBSTYPE = 64

  max number of obs type in RINEX
- const double MAXDTOE = 7200.0

  max time difference to GPS Toe (s)
- const double MAXDTOE_QZS = 7200.0

  max time difference to QZSS Toe (s)
- const double MAXDTOE_GAL = 10800.0

  max time difference to Galileo Toe (s)
- const double MAXDTOE_BDS = 21600.0

  max time difference to BeiDou Toe (s)
- const double MAXDTOE_GLO = 1800.0

  max time difference to GLONASS Toe (s)
- const double MAXDTOE_SBS = 360.0

  max time difference to SBAS Toe (s)
- const double MAXDTOE_S = 86400.0

  max time difference to ephem toe (s) for other
- const double MAXGDOP = 300.0

  max GDOP
- const int MAXSBSURA = 8

  max URA of SBAS satellite
- const int MAXBAND = 10

  max SBAS band of IGP
- const int MAXNIGP = 201

  max number of IGP in SBAS band
- const int MAXNGEO = 4

  max number of GEO satellites
- const int MAXSOLMSG = 8191

  max length of solution message
- const int MAXERRMSG = 4096

  max length of error/warning message
- const int IONOOPT_OFF = 0

  ionosphere option: correction off
- const int IONOOPT_BRDC = 1

  ionosphere option: broadcast model
- const int IONOOPT_SBAS = 2

  ionosphere option: SBAS model
- const int IONOOPT_IFLC = 3

  ionosphere option: L1/L2 or L1/L5 iono-free LC
- const int IONOOPT_EST = 4

  ionosphere option: estimation
- const int IONOOPT_TEC = 5

  ionosphere option: IONEX TEC model

- const int IONOOPT_QZS = 6
  ionosphere option: QZSS broadcast model
- const int IONOOPT_LEX = 7
  ionosphere option: QZSS LEX ionospehre
- const int IONOOPT_STEC = 8
  ionosphere option: SLANT TEC model
- const int TROPOPT_OFF = 0
  troposphere option: correction off
- const int TROPOPT_SAAS = 1
  troposphere option: Saastamoinen model
- const int TROPOPT_SBAS = 2
  troposphere option: SBAS model
- const int TROPOPT_EST = 3
  troposphere option: ZTD estimation
- const int TROPOPT_ESTG = 4
  troposphere option: ZTD+grad estimation
- const int TROPOPT_COR = 5
  troposphere option: ZTD correction
- const int TROPOPT_CORG = 6
  troposphere option: ZTD+grad correction
- const int EPHOPT_BRDC = 0
  ephemeris option: broadcast ephemeris
- const int EPHOPT_PREC = 1
  ephemeris option: precise ephemeris
- const int EPHOPT_SBAS = 2
  ephemeris option: broadcast + SBAS
- const int EPHOPT_SSRAPC = 3
  ephemeris option: broadcast + SSR_APC
- const int EPHOPT_SSRCOM = 4
  ephemeris option: broadcast + SSR_COM
- const int EPHOPT_LEX = 5
  ephemeris option: QZSS LEX ephemeris
- constexpr double EFACT_GPS = 1.0
  error factor: GPS
- constexpr double EFACT_GLO = 1.5
  error factor: GLONASS
- constexpr double EFACT_GAL = 1.0
  error factor: Galileo
- constexpr double EFACT_QZS = 1.0
  error factor: QZSS
- constexpr double EFACT_BDS = 1.0
  error factor: BeiDou
- constexpr double EFACT_IRN = 1.5
  error factor: IRNSS
- constexpr double EFACT_SBS = 3.0
  error factor: SBAS
- const int MAXEXFILE = 1024
  max number of expanded files
- const double MAXSBSAGEF = 30.0
  max age of SBAS fast correction (s)
- const double MAXSBSAGEL = 1800.0

max age of SBAS long term corr (s)

- const int ARMODE_OFF = 0

  AR mode: off.
- const int ARMODE_CONT = 1

  AR mode: continuous.
- const int ARMODE_INST = 2

  AR mode: instantaneous.
- const int ARMODE_FIXHOLD = 3

  AR mode: fix and hold.
- const int ARMODE_PPPAR = 4

  AR mode: PPP-AR.
- const int ARMODE_PPPAR_ILS = 5

  AR mode: AR mode: PPP-AR ILS.
- const int ARMODE_WLNL = 6
- const int ARMODE_TCAR = 7
- const int POSOPT_RINEX = 3

  pos option: rinex header pos
- const int MAXSTRPATH = 1024

  max length of stream path
- const int MAXSTRMSG = 1024

  max length of stream message
- const double CHISQR [100]
- const double LAM_CARR [MAXFREQ]
- const int STRFMT_RTCM2 = 0
- const int STRFMT_RTCM3 = 1
- const int STRFMT_SP3 = 16
- const int STRFMT_RNXCLK = 17
- const int STRFMT_SBAS = 18
- const int STRFMT_NMEA = 19
- const int MAXSTRRTK = 8

13.161.1 Detailed Description

main header file for the rtklib library

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017-2023, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017-2023, Carles Fernandez
All rights reserved.

SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib.h.

13.162 rtklib.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib.h
00003  * \brief main header file for the rtklib library
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017-2023, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017-2023, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * ------------------------------------------------------------------------------
00031  */
00032
00033 #ifndef GNSS_SDR_RTKLIB_H
00034 #define GNSS_SDR_RTKLIB_H
00035
00036 #include "MATH_CONSTANTS.h"
00037 #include "gnss_frequencies.h"
00038 #include "gnss_obs_codes.h"
00039 #include <cctype>
00040 #include <cmath>
00041 #include <cstdarg>
00042 #include <cstdint>
00043 #include <cstdlib>
00044 #include <netinet/in.h>
00045 #include <pthread.h>
00046 #include <string>
00047
00048 /** \addtogroup PVT
00049  * \{ */
00050 /** \addtogroup RTKLIB_Library algorithms_libs_rtklib
00051  * Our version of the RTKLIB core library (see http://www.rtklib.com/)
00052  * \{ */
00053
00054
00055 /* macros --------------------------------------------------------------------*/
00056
00057 #define dev_t int
00058 #define socket_t int
00059 #define closesocket close
00060 #define lock_t pthread_mutex_t
00061 #define initlock(f) pthread_mutex_init(f, NULL)
00062 #define rtk_lock(f) pthread_mutex_lock(f)
00063 #define rtk_unlock(f) pthread_mutex_unlock(f)
00064
00065 #define VER_RTKLIB "2.4.2"
00066 #define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
00067 #define NTRIP_CLI_PORT 2101                     /* default ntrip-client connection port */
00068 #define NTRIP_SVR_PORT 80                       /* default ntrip-server connection port */
00069 #define NTRIP_MAXRSP 32768                      /* max size of ntrip response */
00070 #define NTRIP_MAXSTR 256                        /* max length of mountpoint string */
00071 #define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"       /* ntrip response: client */
00072 #define NTRIP_RSP_OK_SVR "OK\r\n"               /* ntrip response: server */
00073 #define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n" /* ntrip response: source table */
00074 #define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
00075 #define NTRIP_RSP_HTTP "HTTP/"  /* ntrip response: http */
00076 #define NTRIP_RSP_ERROR "ERROR" /* ntrip response: error */
00077
00078 #define FTP_CMD "wget" /* ftp/http command */
00079
00080 const int TINTACT = 200;        //!<  period for stream active (ms)
00081 const int SERIBUFFSIZE = 4096;  //!<  serial buffer size (bytes)
00082 const int TIMETAGH_LEN = 64;    //!<  time tag file header length
00083 const int MAXCLI = 32;          //!<  max client connection for tcp svr
```

```
00084 const int MAXSTATMSG = 32;      //!<  max length of status message
00085
00086 const int FTP_TIMEOUT = 30;  //!< ftp/http timeout (s)
00087 const int MAXRAWLEN = 4096;  //!< max length of receiver raw message
00088 const int MAXSOLBUF = 256;   //!< max number of solution buffer
00089 const int MAXSBSMSG = 32;    //!< max number of SBAS msg in RTK server
00090 const int MAXOBSBUF = 128;   //!< max number of observation data buffer
00091
00092 const int FILEPATHSEP = '/';
00093 const double RE_WGS84 = 6378137.0;              //!< earth semimajor axis (WGS84) (m)
00094 const double FE_WGS84 = (1.0 / 298.257223563);  //!< earth flattening (WGS84)
00095
00096 const double HION = 350000.0;    //!<  ionosphere height (m)
00097 const double PRN_HWBIAS = 1e-6;  //!<  process noise of h/w bias (m/MHz/sqrt(s))
00098
00099 const double INT_SWAP_STAT = 86400.0;  //!<  swap interval of solution status file (s)
00100 const double INT_SWAP_TRAC = 86400.0;  //!<  swap interval of trace file (s)
00101
00102 const unsigned int POLYCRC32 = 0xEDB88320u;  //!<  CRC32 polynomial
00103 const unsigned int POLYCRC24Q = 0x1864CFBu;  //!<  CRC24Q polynomial
00104
00105 const int PMODE_SINGLE = 0;      //!<  positioning mode: single
00106 const int PMODE_DGPS = 1;        //!<  positioning mode: DGPS/DGNSS
00107 const int PMODE_KINEMA = 2;      //!<  positioning mode: kinematic
00108 const int PMODE_STATIC = 3;      //!<  positioning mode: static
00109 const int PMODE_MOVEB = 4;       //!<  positioning mode: moving-base
00110 const int PMODE_FIXED = 5;       //!<  positioning mode: fixed
00111 const int PMODE_PPP_KINEMA = 6;  //!<  positioning mode: PPP-kinemaric
00112 const int PMODE_PPP_STATIC = 7;  //!<  positioning mode: PPP-static
00113 const int PMODE_PPP_FIXED = 8;   //!<  positioning mode: PPP-fixed
00114
00115 const int SOLF_LLH = 0;   //!<  solution format: lat/lon/height
00116 const int SOLF_XYZ = 1;   //!<  solution format: x/y/z-ecef
00117 const int SOLF_ENU = 2;   //!<  solution format: e/n/u-baseline
00118 const int SOLF_NMEA = 3;  //!<  solution format: NMEA-183
00119 const int SOLF_STAT = 4;  //!<  solution format: solution status
00120 const int SOLF_GSIF = 5;  //!<  solution format: GSI F1/F2
00121
00122 const int SOLQ_NONE = 0;    //!<  solution status: no solution
00123 const int SOLQ_FIX = 1;     //!<  solution status: fix
00124 const int SOLQ_FLOAT = 2;   //!<  solution status: float
00125 const int SOLQ_SBAS = 3;    //!<  solution status: SBAS
00126 const int SOLQ_DGPS = 4;    //!<  solution status: DGPS/DGNSS
00127 const int SOLQ_SINGLE = 5;  //!<  solution status: single
00128 const int SOLQ_PPP = 6;     //!<  solution status: PPP
00129 const int SOLQ_DR = 7;      //!<  solution status: dead reckoning
00130 const int MAXSOLQ = 7;      //!<  max number of solution status
00131
00132 const int TIMES_GPST = 0;  //!<  time system: gps time
00133 const int TIMES_UTC = 1;   //!<  time system: utc
00134 const int TIMES_JST = 2;   //!<  time system: jst
00135
00136
00137 const double ERR_SAAS = 0.3;    //!<  saastamoinen model error std (m)
00138 const double ERR_BRDCI = 0.5;   //!<  broadcast iono model error factor
00139 const double ERR_CBIAS = 0.3;   //!<  code bias error std (m)
00140 const double REL_HUMI = 0.7;    //!<  relative humidity for saastamoinen model
00141 const double GAP_RESION = 120;  //!<  default gap to reset ionos parameters (ep)
00142
00143 const int MAXFREQ = 7;  //!<  max NFREQ
00144
00145 const int MAXLEAPS = 64;     //!<  max number of leap seconds table
00146 const double DTTOL = 0.005;  //!<  tolerance of time difference (s)
00147
00148 const int NFREQ = 3;     //!<   number of carrier frequencies
00149 const int NFREQGLO = 2;  //!<   number of carrier frequencies of GLONASS
00150 const int NEXOBS = 0;    //!<  number of extended obs codes
00151 const int MAXANT = 64;   //!<  max length of station name/antenna type
00152
00153 const int MINPRNGPS = 1;                          //!<   min satellite PRN number of GPS
00154 const int MAXPRNGPS = 32;                         //!<   max satellite PRN number of GPS
00155 const int NSATGPS = (MAXPRNGPS - MINPRNGPS + 1);  //!<   number of GPS satellites
00156 const int NSYSGPS = 1;
00157
00158 const int SYS_NONE = 0x00;  //!<   navigation system: none
00159 const int SYS_GPS = 0x01;   //!<   navigation system: GPS
00160 const int SYS_SBS = 0x02;   //!<   navigation system: SBAS
00161 const int SYS_GLO = 0x04;   //!<   navigation system: GLONASS
00162 const int SYS_GAL = 0x08;   //!<   navigation system: Galileo
00163 const int SYS_QZS = 0x10;   //!<   navigation system: QZSS
00164 const int SYS_BDS = 0x20;   //!<   navigation system: BeiDou
00165 const int SYS_IRN = 0x40;   //!<   navigation system: IRNS
00166 const int SYS_LEO = 0x80;   //!<   navigation system: LEO
00167 const int SYS_ALL = 0xFF;   //!<   navigation system: all
00168
00169
00170 #define ENAGLO
```

```
00171 #ifdef ENAGLO
00172 const int MINPRNGLO = 1;                         //!<   min satellite slot number of GLONASS
00173 const int MAXPRNGLO = 27;                        //!<   max satellite slot number of GLONASS
00174 const int NSATGLO = (MAXPRNGLO - MINPRNGLO + 1);  //!<   number of GLONASS satellites
00175 const int NSYSGLO = 1;
00176 #else
00177 const int MINPRNGLO = 0;
00178 const int MAXPRNGLO = 0;
00179 const int NSATGLO = 0;
00180 const int NSYSGLO = 0;
00181 #endif
00182
00183
00184 const int MINPRNGAL = 1;                         //!<   min satellite PRN number of Galileo
00185 const int MAXPRNGAL = 36;                        //!<   max satellite PRN number of Galileo
00186 const int NSATGAL = (MAXPRNGAL - MINPRNGAL + 1);  //!<   number of Galileo satellites
00187 const int NSYSGAL = 1;
00188
00189 const int MAXPRNQZS = 206;  //!<   max satellite PRN number of QZSS
00190 const int MINPRNQZS = 193;  //!<   min satellite PRN number of QZSS
00191 #define ENAQZS
00192 #ifdef ENAQZS
00193 const int MINPRNQZS_S = 183;                     //!<   min satellite PRN number of QZSS SAIF
00194 const int MAXPRNQZS_S = 191;                     //!<   max satellite PRN number of QZSS SAIF
00195 const int NSATQZS = (MAXPRNQZS - MINPRNQZS + 1);  //!<   number of QZSS satellites
00196 const int NSYSQZS = 1;
00197 #else
00198 const int MINPRNQZS_S = 0;
00199 const int NSATQZS = 0;
00200 const int NSYSQZS = 0;
00201 #endif
00202
00203 #define ENABDS
00204 #ifdef ENABDS
00205 const int MINPRNBDS = 1;                         //!<   min satellite sat number of BeiDou
00206 const int MAXPRNBDS = 63;                        //!<   max satellite sat number of BeiDou
00207 const int NSATBDS = (MAXPRNBDS - MINPRNBDS + 1);  //!<   number of BeiDou satellites
00208 const int NSYSBDS = 1;
00209 #else
00210 const int MINPRNBDS = 0;
00211 const int MAXPRNBDS = 0;
00212 const int NSATBDS = 0;
00213 const int NSYSBDS = 0;
00214 #endif
00215
00216 const int MINPRNIRN = 1;  //!<   min satellite sat number of IRNSS
00217 const int MAXPRNIRN = 7;  //!<   max satellite sat number of IRNSS
00218 #ifdef ENAIRN
00219 const int NSATIRN = (MAXPRNIRN - MINPRNIRN + 1);  //!<   number of IRNSS satellites
00220 const int NSYSIRN = 1;
00221 #else
00222 const int NSATIRN = 0;
00223 const int NSYSIRN = 0;
00224 #endif
00225
00226 const int MINPRNLEO = 1;   //!<   min satellite sat number of LEO
00227 const int MAXPRNLEO = 10;  //!<   max satellite sat number of LEO */
00228 #ifdef ENALEO
00229 const int NSATLEO = (MAXPRNLEO - MINPRNLEO + 1);  //!<   number of LEO satellites
00230 const int NSYSLEO = 1;
00231 #else
00232 const int NSATLEO = 0;
00233 const int NSYSLEO = 0;
00234 #endif
00235
00236 const int NSYS = (NSYSGPS + NSYSGLO + NSYSGAL + NSYSQZS + NSYSBDS + NSYSIRN + NSYSLEO);  //!< number
     of systems
00237
00238 const int MINPRNSBS = 120;                       //!<   min satellite PRN number of SBAS
00239 const int MAXPRNSBS = 142;                       //!<   max satellite PRN number of SBAS
00240 const int NSATSBS = (MAXPRNSBS - MINPRNSBS + 1);  //!<   number of SBAS satellites
00241
00242 const int MAXSAT = (NSATGPS + NSATGLO + NSATGAL + NSATQZS + NSATBDS + NSATIRN + NSATSBS + NSATLEO);
00243
00244 const int MAXSTA = 255;
00245
00246 #ifndef MAXOBS
00247 const int MAXOBS = 64;  //!<    max number of obs in an epoch
00248 #endif
00249
00250 const int MAXRCV = 64;               //!<    max receiver number (1 to MAXRCV)
00251 const int MAXOBSTYPE = 64;           //!<    max number of obs type in RINEX
00252 const double MAXDTOE = 7200.0;       //!<    max time difference to GPS Toe (s)
00253 const double MAXDTOE_QZS = 7200.0;   //!<    max time difference to QZSS Toe (s)
00254 const double MAXDTOE_GAL = 10800.0;  //!<    max time difference to Galileo Toe (s)
00255 const double MAXDTOE_BDS = 21600.0;  //!<    max time difference to BeiDou Toe (s)
00256 const double MAXDTOE_GLO = 1800.0;   //!<    max time difference to GLONASS Toe (s)
```

```
00257 const double MAXDTOE_SBS = 360.0;     //!<    max time difference to SBAS Toe (s)
00258 const double MAXDTOE_S = 86400.0;     //!<    max time difference to ephem toe (s) for other
00259 const double MAXGDOP = 300.0;         //!<    max GDOP
00260
00261 const int MAXSBSURA = 8;  //!<    max URA of SBAS satellite
00262 const int MAXBAND = 10;   //!<    max SBAS band of IGP
00263 const int MAXNIGP = 201;  //!<    max number of IGP in SBAS band
00264 const int MAXNGEO = 4;    //!<    max number of GEO satellites
00265
00266 const int MAXSOLMSG = 8191;  //!<    max length of solution message
00267 const int MAXERRMSG = 4096;  //!<    max length of error/warning message
00268
00269 const int IONOOPT_OFF = 0;   //!<    ionosphere option: correction off
00270 const int IONOOPT_BRDC = 1;  //!<    ionosphere option: broadcast model
00271 const int IONOOPT_SBAS = 2;  //!<    ionosphere option: SBAS model
00272 const int IONOOPT_IFLC = 3;  //!<    ionosphere option: L1/L2 or L1/L5 iono-free LC
00273 const int IONOOPT_EST = 4;   //!<    ionosphere option: estimation
00274 const int IONOOPT_TEC = 5;   //!<    ionosphere option: IONEX TEC model
00275 const int IONOOPT_QZS = 6;   //!<    ionosphere option: QZSS broadcast model
00276 const int IONOOPT_LEX = 7;   //!<    ionosphere option: QZSS LEX ionospehre
00277 const int IONOOPT_STEC = 8;  //!<    ionosphere option: SLANT TEC model
00278
00279 const int TROPOPT_OFF = 0;   //!<    troposphere option: correction off
00280 const int TROPOPT_SAAS = 1;  //!<    troposphere option: Saastamoinen model
00281 const int TROPOPT_SBAS = 2;  //!<    troposphere option: SBAS model
00282 const int TROPOPT_EST = 3;   //!<    troposphere option: ZTD estimation
00283 const int TROPOPT_ESTG = 4;  //!<    troposphere option: ZTD+grad estimation
00284 const int TROPOPT_COR = 5;   //!<    troposphere option: ZTD correction
00285 const int TROPOPT_CORG = 6;  //!<    troposphere option: ZTD+grad correction
00286
00287
00288 const int EPHOPT_BRDC = 0;    //!<    ephemeris option: broadcast ephemeris
00289 const int EPHOPT_PREC = 1;    //!<    ephemeris option: precise ephemeris
00290 const int EPHOPT_SBAS = 2;    //!<    ephemeris option: broadcast + SBAS
00291 const int EPHOPT_SSRAPC = 3;  //!<    ephemeris option: broadcast + SSR_APC
00292 const int EPHOPT_SSRCOM = 4;  //!<    ephemeris option: broadcast + SSR_COM
00293 const int EPHOPT_LEX = 5;     //!<    ephemeris option: QZSS LEX ephemeris
00294
00295
00296 constexpr double EFACT_GPS = 1.0;  //!<    error factor: GPS
00297 constexpr double EFACT_GLO = 1.5;  //!<    error factor: GLONASS
00298 constexpr double EFACT_GAL = 1.0;  //!<    error factor: Galileo
00299 constexpr double EFACT_QZS = 1.0;  //!<    error factor: QZSS
00300 constexpr double EFACT_BDS = 1.0;  //!<    error factor: BeiDou
00301 constexpr double EFACT_IRN = 1.5;  //!<    error factor: IRNSS
00302 constexpr double EFACT_SBS = 3.0;  //!<    error factor: SBAS
00303
00304 //! Measurement error factor of a navigation system (single home of the
00305 //! sys-to-EFACT mapping used by the single-point, relative and PPP error
00306 //! models)
00307 inline double sysefact(int sys)
00308 {
00309     switch (sys)
00310         {
00311         case SYS_GLO:
00312             return EFACT_GLO;
00313         case SYS_GAL:
00314             return EFACT_GAL;
00315         case SYS_SBS:
00316             return EFACT_SBS;
00317         case SYS_QZS:
00318             return EFACT_QZS;
00319         case SYS_BDS:
00320             return EFACT_BDS;
00321         case SYS_IRN:
00322             return EFACT_IRN;
00323         case SYS_GPS:
00324         default:
00325             return EFACT_GPS;
00326         }
00327 }
00328
00329 const int MAXEXFILE = 1024;         //!<    max number of expanded files
00330 const double MAXSBSAGEF = 30.0;     //!<    max age of SBAS fast correction (s)
00331 const double MAXSBSAGEL = 1800.0;   //!<    max age of SBAS long term corr (s)
00332
00333 const int ARMODE_OFF = 0;         //!< AR mode: off
00334 const int ARMODE_CONT = 1;        //!< AR mode: continuous
00335 const int ARMODE_INST = 2;        //!< AR mode: instantaneous
00336 const int ARMODE_FIXHOLD = 3;     //!< AR mode: fix and hold
00337 const int ARMODE_PPPAR = 4;       //!< AR mode: PPP-AR
00338 const int ARMODE_PPPAR_ILS = 5;   //!< AR mode: AR mode: PPP-AR ILS
00339 const int ARMODE_WLNL = 6;
00340 const int ARMODE_TCAR = 7;
00341
00342
00343 const int POSOPT_RINEX = 3;    //!< pos option: rinex header pos
```

```
00344 const int MAXSTRPATH = 1024;  //!<  max length of stream path
00345 const int MAXSTRMSG = 1024;   //!<  max length of stream message
00346
00347 using fatalfunc_t = void(const char *);  //!<  fatal callback function type
00348
00349 // clang-format off
00350 #define STR_MODE_R 0x1  /* stream mode: read */
00351 #define STR_MODE_W 0x2  /* stream mode: write */
00352 #define STR_MODE_RW 0x3 /* stream mode: read/write */
00353
00354 #define STR_NONE 0     /* stream type: none */
00355 #define STR_SERIAL 1   /* stream type: serial */
00356 #define STR_FILE 2     /* stream type: file */
00357 #define STR_TCPSVR 3   /* stream type: TCP server */
00358 #define STR_TCPCLI 4   /* stream type: TCP client */
00359 #define STR_UDP 5      /* stream type: UDP stream */
00360 #define STR_NTRIPSVR 6 /* stream type: NTRIP server */
00361 #define STR_NTRIPCLI 7 /* stream type: NTRIP client */
00362 #define STR_FTP 8      /* stream type: ftp */
00363 #define STR_HTTP 9     /* stream type: http */
00364
00365 #define NP_PPP(opt) ((opt)->dynamics ? 9 : 3)
      /* number of pos solution */
00366 #define IC_PPP(s, opt) (NP_PPP(opt) + (s))
      /* state index of clocks (s=0:gps,1:glo) */
00367 #define IT_PPP(opt) (IC_PPP(0, opt) + NSYS)
      /* state index of tropos */
00368 #define NR_PPP(opt) (IT_PPP(opt) + ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt == TROPOPT_EST
      ? 1 : 3))) /* number of solutions */
00369 #define IB_PPP(s, opt) (NR_PPP(opt) + (s)-1)
      /* state index of phase bias */
00370 #define NX_PPP(opt) (IB_PPP(MAXSAT, opt) + 1)
      /* number of estimated states */
00371 // clang-format on
00372
00373 #define NF_RTK(opt) ((opt)->ionoopt == IONOOPT_IFLC ? 1 : (opt)->nf)
00374 #define NP_RTK(opt) ((opt)->dynamics == 0 ? 3 : 9)
00375 #define NI_RTK(opt) ((opt)->ionoopt != IONOOPT_EST ? 0 : MAXSAT)
00376 #define NT_RTK(opt) ((opt)->tropopt < TROPOPT_EST ? 0 : ((opt)->tropopt < TROPOPT_ESTG ? 2 : 6))
00377 #define NL_RTK(opt) ((opt)->glomodear != 2 ? 0 : NFREQGLO)
00378 #define NB_RTK(opt) ((opt)->mode <= PMODE_DGPS ? 0 : MAXSAT * NF_RTK(opt))
00379 #define NR_RTK(opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + NL_RTK(opt))
00380 #define NX_RTK(opt) (NR_RTK(opt) + NB_RTK(opt))
00381
00382 typedef struct
00383 {               /* time struct */
00384     time_t time; /* time (s) expressed by standard time_t */
00385     double sec;  /* fraction of second under 1 s */
00386 } gtime_t;
00387
00388
00389 typedef struct
00390 {                                   /* observation data record */
00391     gtime_t time;                   /* receiver sampling time (GPST) */
00392     unsigned char sat, rcv;         /* satellite/receiver number */
00393     unsigned char SNR[NFREQ + NEXOBS];  /* signal strength (0.25 dBHz) */
00394     unsigned char LLI[NFREQ + NEXOBS];  /* loss of lock indicator */
00395     unsigned char code[NFREQ + NEXOBS]; /* code indicator (CODE_???) */
00396     double L[NFREQ + NEXOBS];           /* observation data carrier-phase (cycle) */
00397     double P[NFREQ + NEXOBS];           /* observation data pseudorange (m) */
00398     float D[NFREQ + NEXOBS];            /* observation data doppler frequency (Hz) */
00399     float Lstd[NFREQ + NEXOBS];         /* receiver-reported stdev of carrier phase (cycle)
      (0:unknown) */
00400     float Pstd[NFREQ + NEXOBS];         /* receiver-reported stdev of pseudorange (m) (0:unknown) */
00401 } obsd_t;
00402
00403
00404 typedef struct
00405 {               /* observation data */
00406     int n, nmax;  /* number of observation data/allocated */
00407     obsd_t *data; /* observation data records */
00408 } obs_t;
00409
00410
00411 typedef struct
00412 {                   /* earth rotation parameter data type */
00413     double mjd;     /* mjd (days) */
00414     double xp, yp;  /* pole offset (rad) */
00415     double xpr, ypr; /* pole offset rate (rad/day) */
00416     double ut1_utc;  /* ut1-utc (s) */
00417     double lod;     /* length of day (s/day) */
00418 } erpd_t;
00419
00420
00421 typedef struct
00422 {               /* earth rotation parameter type */
00423     int n, nmax;  /* number and max number of data */
```

```
00424     erpd_t *data; /* earth rotation parameter data */
00425 } erp_t;
00426
00427
00428 typedef struct
00429 {                         /* antenna parameter type */
00430     int sat;              /* satellite number (0:receiver) */
00431     char type[MAXANT];    /* antenna type */
00432     char code[MAXANT];    /* serial number or satellite code */
00433     gtime_t ts, te;       /* valid time start and end */
00434     double off[NFREQ][3]; /* phase center offset e/n/u or x/y/z (m) */
00435     double var[NFREQ][19]; /* phase center variation (m) */
00436                           /* el=90,85,...,0 or nadir=0,1,2,3,... (deg) */
00437 } pcv_t;
00438
00439
00440 typedef struct
00441 {               /* antenna parameters type */
00442     int n, nmax; /* number of data/allocated */
00443     pcv_t *pcv;  /* antenna parameters data */
00444 } pcvs_t;
00445
00446
00447 typedef struct
00448 {               /* almanac type */
00449     int sat;    /* satellite number */
00450     int svh;    /* sv health (0:ok) */
00451     int svconf; /* as and sv config */
00452     int week;   /* GPS/QZS: gps week, GAL: galileo week */
00453     gtime_t toa; /* Toa */
00454                 /* SV orbit parameters */
00455     double A, e, i0, OMG0, omg, M0, OMGd;
00456     double toas;   /* Toa (s) in week */
00457     double f0, f1; /* SV clock parameters (af0,af1) */
00458 } alm_t;
00459
00460
00461 typedef struct
00462 {                         /* GPS/QZS/GAL broadcast ephemeris type */
00463     int sat;              /* satellite number */
00464     int iode, iodc;       /* IODE,IODC */
00465     int sva;              /* SV accuracy (URA index, GAL: SISA index) */
00466     int svh;              /* SV health (0:ok) */
00467     int week;             /* GPS/QZS: gps week, GAL: galileo week */
00468     int code;             /* GPS/QZS: code on L2, GAL/BDS: data sources */
00469     int flag;             /* GPS/QZS: L2 P data flag, BDS: nav type */
00470     gtime_t toe, toc, ttr; /* Toe,Toc,T_trans */
00471                           /* SV orbit parameters */
00472     double A, e, i0, OMG0, omg, M0, deln, OMGd, idot;
00473     double crc, crs, cuc, cus, cic, cis;
00474     double toes;                          /* Toe (s) in week */
00475     double fit;                           /* fit interval (h) */
00476     double f0, f1, f2;                    /* SV clock parameters (af0,af1,af2) */
00477     double tgd[4];                        /* group delay parameters */
00478                                           /* GPS/QZS:tgd[0]=TGD */
00479                                           /* GAL    :tgd[0]=BGD E5a/E1,tgd[1]=BGD E5b/E1 */
00480                                           /* BDS    :tgd[0]=BGD1,tgd[1]=BGD2 */
00481     double isc[4];                        /* GPS    :isc[0]=ISCL1, isc[1]=ISCL2, isc[2]=ISCL5I,
    isc[3]=ISCL5Q */
00482     double Adot, ndot;                    /* Adot,ndot for CNAV */
00483     float has_clock_correction_m;         /* Galileo High Accuracy Service clock correction, in
    [m] */
00484     float has_orbit_radial_correction_m;  /* Galileo High Accuracy Service orbit radial
    correction, in [m] */
00485     float has_orbit_in_track_correction_m; /* Galileo High Accuracy Service orbit in-track
    correction, in [m] */
00486     float has_orbit_cross_track_correction_m; /* Galileo High Accuracy Service orbit cross-track
    correction, in [m] */
00487     bool apply_has_corrections;
00488     int cnav_uraed;     /* GPS/QZS CNAV URAED index */
00489     int cnav_uraned0;   /* GPS/QZS CNAV URANED0 index */
00490     int cnav_uraned1;   /* GPS/QZS CNAV URANED1 index */
00491     int cnav_uraned2;   /* GPS/QZS CNAV URANED2 index */
00492     double cnav_top;    /* GPS/QZS CNAV CEI data sequence propagation time of week (s) */
00493     int cnav_wnop;      /* GPS/QZS CNAV CEI data sequence propagation week number, modulo 256 (-1 if
    unavailable) */
00494     int cnav_ura_valid; /* GPS/QZS CNAV URA indexes are available */
00495 } eph_t;
00496
00497
00498 typedef struct
00499 {                   /* GLONASS broadcast ephemeris type */
00500     int sat;        /* satellite number */
00501     int iode;       /* IODE (0-6 bit of tb field) */
00502     int frq;        /* satellite frequency number */
00503     int svh, sva, age; /* satellite health, accuracy, age of operation */
00504     gtime_t toe;    /* epoch of epherides (gpst) */
```

```
00505     gtime_t tof;       /* message frame time (gpst) */
00506     double pos[3];     /* satellite position (ecef) (m) */
00507     double vel[3];     /* satellite velocity (ecef) (m/s) */
00508     double acc[3];     /* satellite acceleration (ecef) (m/s^2) */
00509     double taun, gamn; /* SV clock bias (s)/relative freq bias */
00510     double dtaun;      /* delay between L1 and L2 (s) */
00511 } geph_t;
00512
00513
00514 typedef struct
00515 {                          /* precise ephemeris type */
00516     gtime_t time;          /* time (GPST) */
00517     int index;             /* ephemeris index for multiple files */
00518     double pos[MAXSAT][4]; /* satellite position/clock (ecef) (m|s) */
00519     float std[MAXSAT][4];  /* satellite position/clock std (m|s) */
00520     double vel[MAXSAT][4]; /* satellite velocity/clk-rate (m/s|s/s) */
00521     float vst[MAXSAT][4];  /* satellite velocity/clk-rate std (m/s|s/s) */
00522     float cov[MAXSAT][3];  /* satellite position covariance (m^2) */
00523     float vco[MAXSAT][3];  /* satellite velocity covariance (m^2) */
00524 } peph_t;
00525
00526
00527 typedef struct
00528 {                          /* precise clock type */
00529     gtime_t time;          /* time (GPST) */
00530     int index;             /* clock index for multiple files */
00531     double clk[MAXSAT][1]; /* satellite clock (s) */
00532     float std[MAXSAT][1];  /* satellite clock std (s) */
00533 } pclk_t;
00534
00535
00536 typedef struct
00537 {                    /* SBAS ephemeris type */
00538     int sat;         /* satellite number */
00539     gtime_t t0;      /* reference epoch time (GPST) */
00540     gtime_t tof;     /* time of message frame (GPST) */
00541     int sva;         /* SV accuracy (URA index) */
00542     int svh;         /* SV health (0:ok) */
00543     double pos[3];   /* satellite position (m) (ecef) */
00544     double vel[3];   /* satellite velocity (m/s) (ecef) */
00545     double acc[3];   /* satellite acceleration (m/s^2) (ecef) */
00546     double af0, af1; /* satellite clock-offset/drift (s,s/s) */
00547 } seph_t;
00548
00549
00550 typedef struct
00551 {                   /* norad two line element data type */
00552     char name[32];  /* common name */
00553     char alias[32]; /* alias name */
00554     char satno[16]; /* satellite catalog number */
00555     char satclass;  /* classification */
00556     char desig[16]; /* international designator */
00557     gtime_t epoch;  /* element set epoch (UTC) */
00558     double ndot;    /* 1st derivative of mean motion */
00559     double nddot;   /* 2st derivative of mean motion */
00560     double bstar;   /* B* drag term */
00561     int etype;      /* element set type */
00562     int eleno;      /* element number */
00563     double inc;     /* orbit inclination (deg) */
00564     double OMG;     /* right ascension of ascending node (deg) */
00565     double ecc;     /* eccentricity */
00566     double omg;     /* argument of perigee (deg) */
00567     double M;       /* mean anomaly (deg) */
00568     double n;       /* mean motion (rev/day) */
00569     int rev;        /* revolution number at epoch */
00570 } tled_t;
00571
00572
00573 typedef struct
00574 {                 /* norad two line element type */
00575     int n, nmax;  /* number/max number of two line element data */
00576     tled_t *data; /* norad two line element data */
00577 } tle_t;
00578
00579
00580 typedef struct
00581 {                   /* TEC grid type */
00582     gtime_t time;   /* epoch time (GPST) */
00583     int ndata[3];   /* TEC grid data size {nlat,nlon,nhgt} */
00584     double rb;      /* earth radius (km) */
00585     double lats[3]; /* latitude start/interval (deg) */
00586     double lons[3]; /* longitude start/interval (deg) */
00587     double hgts[3]; /* heights start/interval (km) */
00588     double *data;   /* TEC grid data (tecu) */
00589     float *rms;     /* RMS values (tecu) */
00590 } tec_t;
00591
```

```
00592
00593 typedef struct
00594 {                          /* satellite fcb data type */
00595     gtime_t ts, te;         /* start/end time (GPST) */
00596     double bias[MAXSAT][3]; /* fcb value   (cyc) */
00597     double std[MAXSAT][3];  /* fcb std-dev (cyc) */
00598 } fcbd_t;
00599
00600
00601 typedef struct
00602 {                          /* SBAS message type */
00603     int week, tow;          /* reception time */
00604     int prn;                /* SBAS satellite PRN number */
00605     unsigned char msg[29]; /* SBAS message (226bit) padded by 0 */
00606 } sbsmsg_t;
00607
00608
00609 typedef struct
00610 {                   /* SBAS messages type */
00611     int n, nmax;    /* number of SBAS messages/allocated */
00612     sbsmsg_t *msgs; /* SBAS messages */
00613 } sbs_t;
00614
00615
00616 typedef struct
00617 {               /* SBAS fast correction type */
00618     gtime_t t0; /* time of applicability (TOF) */
00619     double prc; /* pseudorange correction (PRC) (m) */
00620     double rrc; /* range-rate correction (RRC) (m/s) */
00621     double dt;  /* range-rate correction delta-time (s) */
00622     int iodf;   /* IODF (issue of date fast corr) */
00623     short udre; /* UDRE+1 */
00624     short ai;   /* degradation factor indicator */
00625 } sbsfcorr_t;
00626
00627
00628 typedef struct
00629 {                    /* SBAS long term satellite error correction type */
00630     gtime_t t0;         /* correction time */
00631     int iode;           /* IODE (issue of date ephemeris) */
00632     double dpos[3];     /* delta position (m) (ecef) */
00633     double dvel[3];     /* delta velocity (m/s) (ecef) */
00634     double daf0, daf1; /* delta clock-offset/drift (s,s/s) */
00635 } sbslcorr_t;
00636
00637
00638 typedef struct
00639 {                     /* SBAS satellite correction type */
00640     int sat;          /* satellite number */
00641     sbsfcorr_t fcorr; /* fast correction */
00642     sbslcorr_t lcorr; /* long term correction */
00643 } sbssatp_t;
00644
00645
00646 typedef struct
00647 {                         /* SBAS satellite corrections type */
00648     int iodp;             /* IODP (issue of date mask) */
00649     int nsat;             /* number of satellites */
00650     int tlat;             /* system latency (s) */
00651     sbssatp_t sat[MAXSAT]; /* satellite correction */
00652 } sbssat_t;
00653
00654
00655 typedef struct
00656 {                   /* SBAS ionospheric correction type */
00657     gtime_t t0;     /* correction time */
00658     short lat, lon; /* latitude/longitude (deg) */
00659     short give;     /* GIVI+1 */
00660     float delay;    /* vertical delay estimate (m) */
00661 } sbsigp_t;
00662
00663
00664 typedef struct
00665 {                     /* IGP band type */
00666     short x;          /* longitude/latitude (deg) */
00667     const short *y;   /* latitudes/longitudes (deg) */
00668     unsigned char bits; /* IGP mask start bit */
00669     unsigned char bite; /* IGP mask end bit */
00670 } sbsigpband_t;
00671
00672
00673 typedef struct
00674 {                         /* SBAS ionospheric corrections type */
00675     int iodi;             /* IODI (issue of date ionos corr) */
00676     int nigp;             /* number of igps */
00677     sbsigp_t igp[MAXNIGP]; /* ionospheric correction */
00678 } sbsion_t;
```

```
00679
00680
00681 typedef struct
00682 {               /* DGPS/GNSS correction type */
00683     gtime_t t0;  /* correction time */
00684     double prc;  /* pseudorange correction (PRC) (m) */
00685     double rrc;  /* range rate correction (RRC) (m/s) */
00686     int iod;     /* issue of data (IOD) */
00687     double udre; /* UDRE */
00688 } dgps_t;
00689
00690
00691 typedef struct
00692 {                           /* SSR correction type */
00693     gtime_t t0[6];          /* epoch time (GPST) {eph,clk,hrclk,ura,bias,pbias} */
00694     double udi[6];          /* SSR update interval (s) */
00695     int iod[6];             /* iod ssr {eph,clk,hrclk,ura,bias,pbias} */
00696     int iode;               /* issue of data */
00697     int iodcrc;             /* issue of data crc for beidou/sbas */
00698     int ura;                /* URA indicator */
00699     int refd;               /* sat ref datum (0:ITRF,1:regional) */
00700     double deph[3];         /* delta orbit {radial,along,cross} (m) */
00701     double ddeph[3];        /* dot delta orbit {radial,along,cross} (m/s) */
00702     double dclk[3];         /* delta clock {c0,c1,c2} (m,m/s,m/s^2) */
00703     double hrclk;           /* high-rate clock correction (m) */
00704     float cbias[MAXCODE];   /* code biases (m) */
00705     double pbias[MAXCODE];  /* phase biases (m) */
00706     float stdpb[MAXCODE];   /* std-dev of phase biases (m) */
00707     double yaw_ang, yaw_rate; /* yaw angle and yaw rate (deg,deg/s) */
00708     unsigned char update;   /* update flag (0:no update,1:update) */
00709 } ssr_t;
00710
00711
00712 typedef struct
00713 {                         /* QZSS LEX message type */
00714     int prn;              /* satellite PRN number */
00715     int type;             /* message type */
00716     int alert;            /* alert flag */
00717     unsigned char stat;   /* signal tracking status */
00718     unsigned char snr;    /* signal C/N0 (0.25 dBHz) */
00719     unsigned int ttt;     /* tracking time (ms) */
00720     unsigned char msg[212]; /* LEX message data part 1695 bits */
00721 } lexmsg_t;
00722
00723
00724 typedef struct
00725 {                   /* QZSS LEX messages type */
00726     int n, nmax;    /* number of LEX messages and allocated */
00727     lexmsg_t *msgs; /* LEX messages */
00728 } lex_t;
00729
00730
00731 typedef struct
00732 {                       /* QZSS LEX ephemeris type */
00733     gtime_t toe;        /* epoch time (GPST) */
00734     gtime_t tof;        /* message frame time (GPST) */
00735     int sat;            /* satellite number */
00736     unsigned char health; /* signal health (L1,L2,L1C,L5,LEX) */
00737     unsigned char ura;  /* URA index */
00738     double pos[3];      /* satellite position (m) */
00739     double vel[3];      /* satellite velocity (m/s) */
00740     double acc[3];      /* satellite acceleration (m/s2) */
00741     double jerk[3];     /* satellite jerk (m/s3) */
00742     double af0, af1;    /* satellite clock bias and drift (s,s/s) */
00743     double tgd;         /* TGD */
00744     double isc[8];      /* ISC */
00745 } lexeph_t;
00746
00747
00748 typedef struct
00749 {                     /* QZSS LEX ionosphere correction type */
00750     gtime_t t0;       /* epoch time (GPST) */
00751     double tspan;     /* valid time span (s) */
00752     double pos0[2];   /* reference position {lat,lon} (rad) */
00753     double coef[3][2]; /* coefficients lat x lon (3 x 2) */
00754 } lexion_t;
00755
00756
00757 typedef struct
00758 {                      /* stec data type */
00759     gtime_t time;      /* time (GPST) */
00760     unsigned char sat; /* satellite number */
00761     double ion;        /* slant ionos delay (m) */
00762     float std;         /* std-dev (m) */
00763     float azel[2];     /* azimuth/elevation (rad) */
00764     unsigned char flag; /* fix flag */
00765 } stec_t;
```

```
00766
00767
00768 typedef struct
00769 {                   /* trop data type */
00770     gtime_t time;   /* time (GPST) */
00771     double trp[3]; /* zenith tropos delay/gradient (m) */
00772     float std[3];  /* std-dev (m) */
00773 } trop_t;
00774
00775
00776 typedef struct
00777 {                                /* ppp corrections type */
00778     int nsta;                    /* number of stations */
00779     char stas[MAXSTA][8];        /* station names */
00780     double rr[MAXSTA][3];        /* station ecef positions (m) */
00781     int ns[MAXSTA], nsmax[MAXSTA]; /* number of stec data */
00782     int nt[MAXSTA], ntmax[MAXSTA]; /* number of trop data */
00783     stec_t *stec[MAXSTA];        /* stec data */
00784     trop_t *trop[MAXSTA];        /* trop data */
00785 } pppcorr_t;
00786
00787
00788 typedef struct
00789 {                               /* navigation data type */
00790     int n, nmax;                /* number of broadcast ephemeris */
00791     int ng, ngmax;              /* number of glonass ephemeris */
00792     int ns, nsmax;              /* number of sbas ephemeris */
00793     int ne, nemax;              /* number of precise ephemeris */
00794     int nc, ncmax;              /* number of precise clock */
00795     int na, namax;              /* number of almanac data */
00796     int nt, ntmax;              /* number of tec grid data */
00797     int nf, nfmax;              /* number of satellite fcb data */
00798     eph_t *eph;                 /* GPS/QZS/GAL ephemeris */
00799     geph_t *geph;               /* GLONASS ephemeris */
00800     seph_t *seph;               /* SBAS ephemeris */
00801     peph_t *peph;               /* precise ephemeris */
00802     pclk_t *pclk;               /* precise clock */
00803     alm_t *alm;                 /* almanac data */
00804     tec_t *tec;                 /* tec grid data */
00805     fcbd_t *fcb;                /* satellite fcb data */
00806     erp_t erp;                  /* earth rotation parameters */
00807     double utc_gps[4];          /* GPS delta-UTC parameters {A0,A1,T,W} */
00808     double utc_glo[4];          /* GLONASS UTC GPS time parameters */
00809     double utc_gal[4];          /* Galileo UTC GPS time parameters */
00810     double utc_qzs[4];          /* QZS UTC GPS time parameters */
00811     double utc_cmp[4];          /* BeiDou UTC parameters */
00812     double utc_irn[4];          /* IRNSS UTC parameters */
00813     double utc_sbs[4];          /* SBAS UTC parameters */
00814     double ion_gps[8];          /* GPS iono model parameters {a0,a1,a2,a3,b0,b1,b2,b3} */
00815     double ion_gal[4];          /* Galileo iono model parameters {ai0,ai1,ai2,0} */
00816     double ion_qzs[8];          /* QZSS iono model parameters {a0,a1,a2,a3,b0,b1,b2,b3} */
00817     double ion_cmp[8];          /* BeiDou DNAV Klobuchar {a0,a1,a2,a3,b0,b1,b2,b3} */
00818     double ion_bdgim[9];        /* BeiDou B-CNAV1 BDGIM alphas alpha1..alpha9 [TECu] */
00819     int ion_bdgim_valid;        /* 1 if ion_bdgim is usable */
00820     double ion_irn[8];          /* IRNSS iono model parameters {a0,a1,a2,a3,b0,b1,b2,b3} */
00821     int leaps;                  /* leap seconds (s) */
00822     double lam[MAXSAT][NFREQ];  /* carrier wave lengths (m) */
00823     double cbias[MAXSAT][3];    /* satellite dcb (0:p1-p2,1:p1-c1,2:p2-c2) (m) */
00824     double rbias[MAXRCV][2][3]; /* receiver dcb (0:p1-p2,1:p1-c1,2:p2-c2) (m) */
00825     double wlbias[MAXSAT];      /* wide-lane bias (cycle) */
00826     double glo_cpbias[4];       /* glonass code-phase bias {1C,1P,2C,2P} (m) */
00827     char glo_fcn[MAXPRNGLO + 1]; /* glonass frequency channel number + 8 */
00828     pcv_t pcvs[MAXSAT];         /* satellite antenna pcv */
00829     sbssat_t sbssat;            /* SBAS satellite corrections */
00830     sbsion_t sbsion[MAXBAND + 1]; /* SBAS ionosphere corrections */
00831     dgps_t dgps[MAXSAT];        /* DGPS corrections */
00832     ssr_t ssr[MAXSAT];          /* SSR corrections */
00833     lexeph_t lexeph[MAXSAT];    /* LEX ephemeris */
00834     lexion_t lexion;            /* LEX ionosphere correction */
00835     pppcorr_t pppcorr;          /* ppp corrections */
00836 } nav_t;
00837
00838
00839 typedef struct
00840 {                       /* station parameter type */
00841     char name[MAXANT];  /* marker name */
00842     char marker[MAXANT]; /* marker number */
00843     char antdes[MAXANT]; /* antenna descriptor */
00844     char antsno[MAXANT]; /* antenna serial number */
00845     char rectype[MAXANT]; /* receiver type descriptor */
00846     char recver[MAXANT]; /* receiver firmware version */
00847     char recsno[MAXANT]; /* receiver serial number */
00848     int antsetup;       /* antenna setup id */
00849     int itrf;           /* ITRF realization year */
00850     int deltype;        /* antenna delta type (0:enu,1:xyz) */
00851     double pos[3];      /* station position (ecef) (m) */
00852     double del[3];      /* antenna position delta (e/n/u or x/y/z) (m) */
```

```
00853     double hgt;           /* antenna height (m) */
00854 } sta_t;
00855
00856
00857 typedef struct
00858 {                         /* solution type */
00859     gtime_t time;         /* time (GPST) */
00860     double rr[6];         /* position/velocity (m|m/s) */
00861                           /* {x,y,z,vx,vy,vz} or {e,n,u,ve,vn,vu} */
00862     float qr[6];          /* position variance/covariance (m^2) */
00863                           /* {c_xx,c_yy,c_zz,c_xy,c_yz,c_zx} or */
00864                           /* {c_ee,c_nn,c_uu,c_en,c_nu,c_ue} */
00865     double dtr[6];        /* receiver clock bias to time systems (s) */
00866     unsigned char type; /* type (0:xyz-ecef,1:enu-baseline) */
00867     unsigned char stat; /* solution status (SOLQ_???) */
00868     unsigned char ns;   /* number of valid satellites */
00869     float age;          /* age of differential (s) */
00870     float ratio;        /* AR ratio factor for validation */
00871     float thres;        /* AR ratio threshold for validation */
00872     float prev_ratio1;  /* previous AR ratio factor for validation */
00873     float prev_ratio2;  /* previous AR ratio factor for validation, after sat exclusion */
00874 } sol_t;
00875
00876
00877 typedef struct
00878 {                                   /* solution buffer type */
00879     int n, nmax;                    /* number of solution/max number of buffer */
00880     int cyclic;                     /* cyclic buffer flag */
00881     int start, end;                 /* start/end index */
00882     gtime_t time;                   /* current solution time */
00883     sol_t *data;                    /* solution data */
00884     double rb[3];                   /* reference position {x,y,z} (ecef) (m) */
00885     unsigned char buff[MAXSOLMSG + 1]; /* message buffer */
00886     int nb;                         /* number of byte in message buffer */
00887 } solbuf_t;
00888
00889
00890 typedef struct
00891 {                         /* solution status type */
00892     gtime_t time;         /* time (GPST) */
00893     unsigned char sat;    /* satellite number */
00894     unsigned char frq;    /* frequency (1:L1,2:L2,...) */
00895     float az, el;         /* azimuth/elevation angle (rad) */
00896     float resp;           /* pseudorange residual (m) */
00897     float resc;           /* carrier-phase residual (m) */
00898     unsigned char flag;   /* flags: (vsat<<5)+(slip<<3)+fix */
00899     unsigned char snr;    /* signal strength (0.25 dBHz) */
00900     unsigned short lock;  /* lock counter */
00901     unsigned short outc;  /* outage counter */
00902     unsigned short slipc; /* slip counter */
00903     unsigned short rejc;  /* reject counter */
00904 } solstat_t;
00905
00906
00907 typedef struct
00908 {                   /* solution status buffer type */
00909     int n, nmax;    /* number of solution/max number of buffer */
00910     solstat_t *data; /* solution status data */
00911 } solstatbuf_t;
00912
00913
00914 typedef struct
00915 {                                           /* RTCM control struct type */
00916     int staid;                              /* station id */
00917     int stah;                               /* station health */
00918     int seqno;                              /* sequence number for rtcm 2 or iods msm */
00919     int outtype;                            /* output message type */
00920     gtime_t time;                           /* message time */
00921     gtime_t time_s;                         /* message start time */
00922     obs_t obs;                              /* observation data (uncorrected) */
00923     nav_t nav;                              /* satellite ephemerides */
00924     sta_t sta;                              /* station parameters */
00925     dgps_t *dgps;                           /* output of dgps corrections */
00926     ssr_t ssr[MAXSAT];                      /* output of ssr corrections */
00927     char msg[128];                          /* special message */
00928     char msgtype[256];                      /* last message type */
00929     char msmtype[6][128];                   /* msm signal types */
00930     int obsflag;                            /* obs data complete flag (1:ok,0:not complete) */
00931     int ephsat;                             /* update satellite of ephemeris */
00932     double cp[MAXSAT][NFREQ + NEXOBS];      /* carrier-phase measurement */
00933     unsigned short lock[MAXSAT][NFREQ + NEXOBS]; /* lock time */
00934     unsigned short loss[MAXSAT][NFREQ + NEXOBS]; /* loss of lock count */
00935     gtime_t lltime[MAXSAT][NFREQ + NEXOBS]; /* last lock time */
00936     int nbyte;                              /* number of bytes in message buffer */
00937     int nbit;                               /* number of bits in word buffer */
00938     int len;                                /* message length (bytes) */
00939     unsigned char buff[1200];               /* message buffer */
```

```
00940     unsigned int word;                          /* word buffer for rtcm 2 */
00941     unsigned int nmsg2[100];                    /* message count of RTCM 2 (1-99:1-99,0:other) */
00942     unsigned int nmsg3[400];                    /* message count of RTCM 3
      (1-299:1001-1299,300-399:2000-2099,0:other) */
00943     char opt[256];                              /* RTCM dependent options */
00944 } rtcm_t;
00945
00946
00947 typedef struct
00948 {                   /* download url type */
00949     char type[32];  /* data type */
00950     char path[1024]; /* url path */
00951     char dir[1024];  /* local directory */
00952     double tint;     /* time interval (s) */
00953 } url_t;
00954
00955
00956 typedef struct
00957 {                       /* option type */
00958     const char *name;   /* option name */
00959     int format;         /* option format (0:int,1:double,2:string,3:enum) */
00960     void *var;          /* pointer to option variable */
00961     const char *comment; /* option comment/enum labels/unit */
00962 } opt_t;
00963
00964
00965 typedef struct
00966 {                           /* extended receiver error model */
00967     int ena[4];             /* model enabled */
00968     double cerr[4][NFREQ * 2]; /* code errors (m) */
00969     double perr[4][NFREQ * 2]; /* carrier-phase errors (m) */
00970     double gpsglob[NFREQ];  /* gps-glonass h/w bias (m) */
00971     double gloicb[NFREQ];   /* glonass interchannel bias (m/fn) */
00972 } exterr_t;
00973
00974
00975 typedef struct
00976 {                       /* SNR mask type */
00977     int ena[2];         /* enable flag {rover,base} */
00978     double mask[NFREQ][9]; /* mask (dBHz) at 5,10,...85 deg */
00979 } snrmask_t;
00980
00981
00982 typedef struct
00983 {                               /* processing options type */
00984     int mode;                   /* positioning mode (PMODE_???) */
00985     int soltype;                /* solution type (0:forward,1:backward,2:combined) */
00986     int nf;                     /* number of frequencies (1:L1,2:L1+L2,3:L1+L2+L5) */
00987     int navsys;                 /* navigation system */
00988     double elmin;               /* elevation mask angle (rad) */
00989     snrmask_t snrmask;          /* SNR mask */
00990     int sateph;                 /* satellite ephemeris/clock (EPHOPT_???) */
00991     int modear;                 /* AR mode (0:off,1:continuous,2:instantaneous,3:fix and
      hold,4:ppp-ar) */
00992     int glomodear;              /* GLONASS AR mode (0:off,1:on,2:auto cal,3:ext cal) */
00993     int bdsmodear;              /* BeiDou AR mode (0:off,1:on) */
00994     int maxout;                 /* obs outage count to reset bias */
00995     int minlock;                /* min lock count to fix ambiguity */
00996     int minfix;                 /* min fix count to hold ambiguity */
00997     int armaxiter;              /* max iteration to resolve ambiguity */
00998     int ionoopt;                /* ionosphere option (IONOOPT_???) */
00999     int tropopt;                /* troposphere option (TROPOPT_???) */
01000     int dynamics;               /* dynamics model (0:none,1:velociy,2:accel) */
01001     int tidecorr;               /* earth tide correction (0:off,1:solid,2:solid+otl+pole) */
01002     int niter;                  /* number of filter iteration */
01003     int codesmooth;             /* code smoothing window size (0:none) */
01004     int intpref;                /* interpolate reference obs (for post mission) */
01005     int sbascorr;               /* SBAS correction options */
01006     int sbassatsel;             /* SBAS satellite selection (0:all) */
01007     int rovpos;                 /* rover position for fixed mode */
01008     int refpos;                 /* base position for relative mode */
01009                                 /* (0:pos in prcopt,  1:average of single pos, */
01010                                 /*  2:read from file, 3:rinex header, 4:rtcm pos) */
01011     double eratio[NFREQ];       /* code/phase error ratio */
01012     double err[8];              /* measurement error factor */
01013                                 /* [5]: SNR reference (dBHz) for the SNR term */
01014                                 /* [6]: SNR error term (m) (0:disabled) */
01015                                 /* [7]: receiver-stdev error term (0:disabled) */
01016                                 /* [0]:reserved */
01017                                 /* [1-3]:error factor a/b/c of phase (m) */
01018                                 /* [4]:doppler frequency (hz) */
01019     double std[3];              /* initial-state std [0]bias,[1]iono [2]trop */
01020     double prn[6];              /* process-noise std [0]bias,[1]iono [2]trop [3]acch [4]accv [5] pos
      */
01021     double sclkstab;            /* satellite clock stability (sec/sec) */
01022     double thresar[8];          /* AR validation threshold */
01023     double elmaskar;            /* elevation mask of AR for rising satellite (deg) */
```

```
01024     double elmaskhold;            /* elevation mask to hold ambiguity (deg) */
01025     double thresslip;             /* slip threshold of geometry-free phase (m) */
01026     double thresdop;              /* slip threshold of doppler (m/s) (0:disabled) */
01027     double maxtdiff;              /* max difference of time (sec) */
01028     double maxinno[2];            /* reject threshold of innovation {phase, code} (m) */
01029     double maxgdop;               /* reject threshold of gdop */
01030     double baseline[2];           /* baseline length constraint {const,sigma} (m) */
01031     double ru[3];                 /* rover position for fixed mode {x,y,z} (ecef) (m) */
01032     double rb[3];                 /* base position for relative mode {x,y,z} (ecef) (m) */
01033     char anttype[2][MAXANT];      /* antenna types {rover,base} */
01034     double antdel[2][3];          /* antenna delta {{rov_e,rov_n,rov_u},{ref_e,ref_n,ref_u}} */
01035     pcv_t pcvr[2];                /* receiver antenna parameters {rov,base} */
01036     unsigned char exsats[MAXSAT]; /* excluded satellites (1:excluded,2:included) */
01037     int maxaveep;                 /* max averaging epoches */
01038     int initrst;                  /* initialize by restart */
01039     int outsingle;                /* output single by dgps/float/fix/ppp outage */
01040     char rnxopt[2][256];          /* rinex options {rover,base} */
01041     int posopt[6];                /* positioning options */
01042     int syncsol;                  /* solution sync mode (0:off,1:on) */
01043     double odisp[2][6 * 11];      /* ocean tide loading parameters {rov,base} */
01044     exterr_t exterr;              /* extended receiver error model */
01045     int freqopt;                  /* disable L2-AR */
01046     char pppopt[256];             /* ppp option */
01047     bool bancroft_init;           /* enable Bancroft initialization for the first iteration of the PVT
      computation */
01048     bool estqzsisb;               /* estimate a separate QZS-GPS inter-system bias in single-point
      positioning.
01049                                      Requires one extra satellite in mixed GPS+QZSS epochs; keep
      disabled in
01050                                      degraded-visibility scenarios */
01051     /* AR management options from the demo5 RTKLIB fork. Value-initialized (all zeros)
01052        they reproduce the classic behavior: no AR filtering, no minimum-satellite gates,
01053        no satellite exclusion cycling, no position-variance gate. PRCOPT_DEFAULT keeps
01054        them at zero; note that the GNSS-SDR PVT adapter enables the demo5 defaults
01055        instead (PVT.ar_filter=true, min_fix_sats=4, min_hold_sats=5, min_drop_sats=10,
01056        ar_max_position_variance=0.25) - set those configuration options to zero/false
01057        to recover the classic behavior */
01058     int arfilter;       /* AR filtering to reject newly-added sats on AR ratio degradation
      (0:off,1:on) */
01059     int minfixsats;     /* min number of sats to fix integer ambiguities (0:no limit) */
01060     int minholdsats;    /* min number of sats to hold integer ambiguities (0:no limit) */
01061     int mindropsats;    /* min number of sats to enable single-sat exclusion cycling in AR
      (0:disabled) */
01062     double varholdamb;  /* variance of the fix-and-hold pseudo measurements (cycle^2) (0:use default)
      */
01063     double armaxposvar; /* max position variance to attempt AR (m^2) (0:no gate) */
01064 } prcopt_t;
01065
01066
01067 typedef struct
01068 {                       /* solution options type */
01069     int posf;           /* solution format (SOLF_???) */
01070     int times;          /* time system (TIMES_???) */
01071     int timef;          /* time format (0:ssss.s,1:yyyy/mm/dd hh:mm:ss.s) */
01072     int timeu;          /* time digits under decimal point */
01073     int degf;           /* latitude/longitude format (0:ddd.ddd,1:ddd mm ss) */
01074     int outhead;        /* output header (0:no,1:yes) */
01075     int outopt;         /* output processing options (0:no,1:yes) */
01076     int datum;          /* datum (0:WGS84,1:Tokyo) */
01077     int height;         /* height (0:ellipsoidal,1:geodetic) */
01078     int geoid;          /* geoid model (0:EGM96,1:JGD2000) */
01079     int solstatic;      /* solution of static mode (0:all,1:single) */
01080     int sstat;          /* solution statistics level (0:off,1:states,2:residuals) */
01081     int trace;          /* debug trace level (0:off,1-5:debug) */
01082     double nmeaintv[2]; /* nmea output interval (s) (<0:no,0:all) */
01083                         /* nmeaintv[0]:gprmc,gpgga,nmeaintv[1]:gpgsv */
01084     char sep[64];       /* field separator */
01085     char prog[64];      /* program name */
01086     double maxsolstd;   /* max std-dev for solution output (m) (0:all) */
01087 } solopt_t;
01088
01089
01090 typedef struct
01091 {                           /* satellite status type */
01092     unsigned char sys;      /* navigation system */
01093     unsigned char vs;       /* valid satellite flag single */
01094     double azel[2];         /* azimuth/elevation angles {az,el} (rad) */
01095     double resp[NFREQ];     /* residuals of pseudorange (m) */
01096     double resc[NFREQ];     /* residuals of carrier-phase (m) */
01097     unsigned char vsat[NFREQ]; /* valid satellite flag */
01098     unsigned char snr[NFREQ];  /* signal strength (0.25 dBHz) */
01099     unsigned char code[NFREQ]; /* code indicator (CODE_???) */
01100     unsigned char fix[NFREQ];  /* ambiguity fix flag (1:fix,2:float,3:hold) */
01101     unsigned char slip[NFREQ]; /* cycle-slip flag */
01102     unsigned char half[NFREQ]; /* half-cycle valid flag */
01103     int lock[NFREQ];           /* lock counter of phase */
01104     unsigned int outc[NFREQ];  /* obs outage counter of phase */
```

```
01105     unsigned int slipc[NFREQ]; /* cycle-slip counter */
01106     unsigned int rejc[NFREQ];  /* reject counter */
01107     double gf;                 /* geometry-free phase L1-L2 (m) */
01108     double gf2;                /* geometry-free phase L1-L5 (m) */
01109     double mw;                 /* MW-LC (m) */
01110     double phw;                /* phase windup (cycle) */
01111     gtime_t pt[2][NFREQ];      /* previous carrier-phase time */
01112     double ph[2][NFREQ];       /* previous carrier-phase observable (cycle) */
01113     float snr_rover[NFREQ];    /* rover signal strength (dBHz) */
01114     float snr_base[NFREQ];     /* base signal strength (dBHz) */
01115 } ssat_t;
01116
01117
01118 typedef struct
01119 {                       /* ambiguity control type */
01120     gtime_t epoch[4];   /* last epoch */
01121     int n[4];           /* number of epochs */
01122     double LC[4];       /* linear combination average */
01123     double LCv[4];      /* linear combination variance */
01124     int fixcnt;         /* fix count */
01125     char flags[MAXSAT]; /* fix flags */
01126 } ambc_t;
01127
01128
01129 typedef struct
01130 {                           /* RTK control/result type */
01131     sol_t sol;              /* RTK solution */
01132     double rb[6];           /* base position/velocity (ecef) (m|m/s) */
01133     int nx, na;             /* number of float states/fixed states */
01134     double tt;              /* time difference between current and previous (s) */
01135     double *x, *P;          /* float states and their covariance */
01136     double *xa, *Pa;        /* fixed states and their covariance */
01137     int nfix;               /* number of continuous fixes of ambiguity */
01138     int excsat;             /* satellite currently excluded from AR for partial ambiguity resolution
      */
01139     int nb_ar;              /* number of double-differenced ambiguities used in the last AR attempt */
01140     int holdamb;            /* set to 1 once fix-and-hold has been applied */
01141     ambc_t ambc[MAXSAT];    /* ambiguity control */
01142     ssat_t ssat[MAXSAT];    /* satellite status */
01143     int neb;                /* bytes in error message buffer */
01144     char errbuf[MAXERRMSG]; /* error message buffer */
01145     prcopt_t opt;           /* processing options */
01146 } rtk_t;
01147
01148
01149 typedef struct half_cyc_tag
01150 {                           /* half-cycle correction list type */
01151     unsigned char sat;      /* satellite number */
01152     unsigned char freq;     /* frequency number (0:L1,1:L2,2:L5) */
01153     unsigned char valid;    /* half-cycle valid flag */
01154     char corr;              /* half-cycle corrected (x 0.5 cyc) */
01155     gtime_t ts, te;         /* time start, time end */
01156     struct half_cyc_tag *next; /* pointer to next correction */
01157 } half_cyc_t;
01158
01159
01160 typedef struct
01161 {                           /* stream type */
01162     int type;               /* type (STR_???) */
01163     int mode;               /* mode (STR_MODE_?) */
01164     int state;              /* state (-1:error,0:close,1:open) */
01165     unsigned int inb, inr;  /* input bytes/rate */
01166     unsigned int outb, outr; /* output bytes/rate */
01167     unsigned int tick, tact; /* tick/active tick */
01168     unsigned int inbt, outbt; /* input/output bytes at tick */
01169     lock_t lock;            /* lock flag */
01170     void *port;             /* type dependent port control struct */
01171     char path[MAXSTRPATH];  /* stream path */
01172     char msg[MAXSTRMSG];    /* stream message */
01173 } stream_t;
01174
01175
01176 typedef struct
01177 {             /* serial control type */
01178     dev_t dev; /* serial device */
01179     int error; /* error state */
01180 } serial_t;
01181
01182
01183 struct file_t
01184 {                           /* file control type */
01185     FILE *fp = nullptr;     /* file pointer */
01186     FILE *fp_tag = nullptr; /* file pointer of tag file */
01187     FILE *fp_tmp = nullptr; /* temporary file pointer for swap */
01188     FILE *fp_tag_tmp = nullptr; /* temporary file pointer of tag file for swap */
01189     std::string path;       /* file path */
01190     std::string openpath;   /* open file path */
```

```
01191     int mode = 0;               /* file mode */
01192     int timetag;                /* time tag flag (0:off,1:on) */
01193     int repmode = 0;            /* replay mode (0:master,1:slave) */
01194     int offset = 0;             /* time offset (ms) for slave */
01195     gtime_t time = {};          /* start time */
01196     gtime_t wtime = {};         /* write time */
01197     unsigned int tick = 0;      /* start tick */
01198     unsigned int tick_f = 0;    /* start tick in file */
01199     unsigned int fpos = 0;      /* current file position */
01200     double start = 0;           /* start offset (s) */
01201     double speed = 0;           /* replay speed (time factor) */
01202     double swapintv = 0;        /* swap interval (hr) (0: no swap) */
01203     lock_t lock;                /* lock flag */
01204 };
01205
01206
01207 typedef struct
01208 {                           /* tcp control type */
01209     int state;              /* state (0:close,1:wait,2:connect) */
01210     char saddr[256];        /* address string */
01211     int port;               /* port */
01212     struct sockaddr_in addr; /* address resolved */
01213     socket_t sock;          /* socket descriptor */
01214     int tcon;               /* reconnect time (ms) (-1:never,0:now) */
01215     unsigned int tact;      /* data active tick */
01216     unsigned int tdis;      /* disconnect tick */
01217 } tcp_t;
01218
01219
01220 typedef struct
01221 {                       /* tcp server type */
01222     tcp_t svr;          /* tcp server control */
01223     tcp_t cli[MAXCLI]; /* tcp client controls */
01224 } tcpsvr_t;
01225
01226
01227 typedef struct
01228 {                /* tcp client type */
01229     tcp_t svr;   /* tcp server control */
01230     int toinact; /* inactive timeout (ms) (0:no timeout) */
01231     int tirecon; /* reconnect interval (ms) (0:no reconnect) */
01232 } tcpcli_t;
01233
01234
01235 typedef struct
01236 {                                           /* ntrip control type */
01237     int state;                              /* state (0:close,1:wait,2:connect) */
01238     int type;                               /* type (0:server,1:client) */
01239     int version;                            /* NTRIP client version (1 or 2) */
01240     int tls_enabled;                        /* TLS transport enabled */
01241     int chunked;                            /* HTTP chunked transfer encoding */
01242     int chunk_state;                        /* chunk parser state (0:size,1:data,2:crlf) */
01243     unsigned int chunk_remaining;           /* bytes left in current HTTP chunk */
01244     int nb;                                 /* response buffer size */
01245     int request_length;                     /* buffered NTRIP request size */
01246     int request_offset;                     /* bytes of buffered request already sent */
01247     int request_sent;                       /* complete request sent on current connection */
01248     int response_received;                  /* response bytes received on current connection */
01249     int v1_retry_requested;                 /* response indicates a fresh v1 retry */
01250     char url[256];                          /* url for proxy */
01251     char host[256];                         /* caster hostname (not proxy hostname) */
01252     char port[16];                          /* caster port */
01253     char mntpnt[256];                       /* mountpoint */
01254     char user[256];                         /* user */
01255     char passwd[256];                       /* password */
01256     char str[NTRIP_MAXSTR];                 /* mountpoint string for server */
01257     unsigned char buff[NTRIP_MAXRSP];       /* response buffer */
01258     unsigned char request_buff[NTRIP_MAXRSP]; /* persistent non-blocking request buffer */
01259     tcpcli_t *tcp;                          /* tcp client */
01260     void *tls_ctx;                          /* opaque Rtklib_Tls_Client pointer */
01261 } ntrip_t;
01262
01263
01264 typedef struct
01265 {              /* ftp download control type */
01266     int state; /* state (0:close,1:download,2:complete,3:error) */
01267     int proto; /* protocol (0:ftp,1:http) */
01268     int error; /* error code (0:no error,1-10:wget error, */
01269     /*            11:no temp dir,12:uncompact error) */
01270     char addr[1024];  /* download address */
01271     char file[1024];  /* download file path */
01272     char user[256];   /* user for ftp */
01273     char passwd[256]; /* password for ftp */
01274     char local[1024]; /* local file path */
01275     int topts[4];     /* time options {poff,tint,toff,tretry} (s) */
01276     gtime_t tnext;    /* next retry time (gpst) */
01277     pthread_t thread; /* download thread */
```

```
01278 } ftp_t;
01279
01280
01281 typedef struct
01282 {                                           /* receiver raw data control type */
01283     gtime_t time;                           /* message time */
01284     gtime_t tobs;                           /* observation data time */
01285     obs_t obs;                              /* observation data */
01286     obs_t obuf;                             /* observation data buffer */
01287     nav_t nav;                              /* satellite ephemerides */
01288     sta_t sta;                              /* station parameters */
01289     int ephsat;                             /* sat number of update ephemeris (0:no satellite) */
01290     sbsmsg_t sbsmsg;                        /* SBAS message */
01291     char msgtype[256];                      /* last message type */
01292     unsigned char subfrm[MAXSAT][380];      /* subframe buffer */
01293     lexmsg_t lexmsg;                        /* LEX message */
01294     double lockt[MAXSAT][NFREQ + NEXOBS];   /* lock time (s) */
01295     double icpp[MAXSAT], off[MAXSAT], icpc; /* carrier params for ss2 */
01296     double prCA[MAXSAT], dpCA[MAXSAT];      /* L1/CA pseudrange/doppler for javad */
01297     unsigned char halfc[MAXSAT][NFREQ + NEXOBS]; /* half-cycle add flag */
01298     char freqn[MAXOBS];                     /* frequency number for javad */
01299     int nbyte;                              /* number of bytes in message buffer */
01300     int len;                                /* message length (bytes) */
01301     int iod;                                /* issue of data */
01302     int tod;                                /* time of day (ms) */
01303     int tbase;                              /* time base (0:gpst,1:utc(usno),2:glonass,3:utc(su)
      */
01304     int flag;                               /* general purpose flag */
01305     int outtype;                            /* output message type */
01306     unsigned char buff[MAXRAWLEN];          /* message buffer */
01307     char opt[256];                          /* receiver dependent options */
01308     double receive_time;                    /* RT17: Reiceve time of week for week rollover
      detection */
01309     unsigned int plen;                      /* RT17: Total size of packet to be read */
01310     unsigned int pbyte;                     /* RT17: How many packet bytes have been read so far
      */
01311     unsigned int page;                      /* RT17: Last page number */
01312     unsigned int reply;                     /* RT17: Current reply number */
01313     int week;                               /* RT17: week number */
01314     unsigned char pbuff[255 + 4 + 2];       /* RT17: Packet buffer */
01315 } raw_t;
01316
01317
01318 typedef struct
01319 {                               /* RTK server type */
01320     int state;                  /* server state (0:stop,1:running) */
01321     int cycle;                  /* processing cycle (ms) */
01322     int nmeacycle;              /* NMEA request cycle (ms) (0:no req) */
01323     int nmeareq;                /* NMEA request (0:no,1:nmeapos,2:single sol) */
01324     double nmeapos[3];          /* NMEA request position (ecef) (m) */
01325     int buffsize;               /* input buffer size (bytes) */
01326     int format[3];              /* input format {rov,base,corr} */
01327     solopt_t solopt[2];         /* output solution options {sol1,sol2} */
01328     int navsel;                 /* ephemeris select (0:all,1:rover,2:base,3:corr) */
01329     int nsbs;                   /* number of sbas message */
01330     int nsol;                   /* number of solution buffer */
01331     rtk_t rtk;                  /* RTK control/result struct */
01332     int nb[3];                  /* bytes in input buffers {rov,base} */
01333     int nsb[2];                 /* bytes in soulution buffers */
01334     int npb[3];                 /* bytes in input peek buffers */
01335     unsigned char *buff[3];     /* input buffers {rov,base,corr} */
01336     unsigned char *sbuf[2];     /* output buffers {sol1,sol2} */
01337     unsigned char *pbuf[3];     /* peek buffers {rov,base,corr} */
01338     sol_t solbuf[MAXSOLBUF];    /* solution buffer */
01339     unsigned int nmsg[3][10];   /* input message counts */
01340     raw_t raw[3];               /* receiver raw control {rov,base,corr} */
01341     rtcm_t rtcm[3];             /* RTCM control {rov,base,corr} */
01342     gtime_t ftime[3];           /* download time {rov,base,corr} */
01343     char files[3][MAXSTRPATH];  /* download paths {rov,base,corr} */
01344     obs_t obs[3][MAXOBSBUF];    /* observation data {rov,base,corr} */
01345     nav_t nav;                  /* navigation data */
01346     sbsmsg_t sbsmsg[MAXSBSMSG]; /* SBAS message buffer */
01347     stream_t stream[8];         /* streams {rov,base,corr,sol1,sol2,logr,logb,logc} */
01348     stream_t *moni;             /* monitor stream */
01349     unsigned int tick;          /* start tick */
01350     pthread_t thread;           /* server thread */
01351     int cputime;                /* CPU time (ms) for a processing cycle */
01352     int prcout;                 /* missing observation data count */
01353     lock_t lock;                /* lock flag */
01354 } rtksvr_t;
01355
01356 typedef struct
01357 {                               /* multi-signal-message header type */
01358     unsigned char iod;          /* issue of data station */
01359     unsigned char time_s;       /* cumulative session transmitting time */
01360     unsigned char clk_str;      /* clock steering indicator */
01361     unsigned char clk_ext;      /* external clock indicator */
```

```
01362     unsigned char smooth;       /* divergence free smoothing indicator */
01363     unsigned char tint_s;       /* soothing interval */
01364     unsigned char nsat, nsig;   /* number of satellites/signals */
01365     unsigned char sats[64];     /* satellites */
01366     unsigned char sigs[32];     /* signals */
01367     unsigned char cellmask[64]; /* cell mask */
01368 } msm_h_t;
01369
01370
01371 const double CHISQR[100] = {/* chi-sqr(n) (alpha=0.001) */
01372     10.8, 13.8, 16.3, 18.5, 20.5, 22.5, 24.3, 26.1, 27.9, 29.6,
01373     31.3, 32.9, 34.5, 36.1, 37.7, 39.3, 40.8, 42.3, 43.8, 45.3,
01374     46.8, 48.3, 49.7, 51.2, 52.6, 54.1, 55.5, 56.9, 58.3, 59.7,
01375     61.1, 62.5, 63.9, 65.2, 66.6, 68.0, 69.3, 70.7, 72.1, 73.4,
01376     74.7, 76.0, 77.3, 78.6, 80.0, 81.3, 82.6, 84.0, 85.4, 86.7,
01377     88.0, 89.3, 90.6, 91.9, 93.3, 94.7, 96.0, 97.4, 98.7, 100,
01378     101, 102, 103, 104, 105, 107, 108, 109, 110, 112,
01379     113, 114, 115, 116, 118, 119, 120, 122, 123, 125,
01380     126, 127, 128, 129, 131, 132, 133, 134, 135, 137,
01381     138, 139, 140, 142, 143, 144, 145, 147, 148, 149};
01382
01383
01384 const double LAM_CARR[MAXFREQ] = {/* carrier wave length (m) */
01385     SPEED_OF_LIGHT_M_S / FREQ1, SPEED_OF_LIGHT_M_S / FREQ2, SPEED_OF_LIGHT_M_S / FREQ5,
      SPEED_OF_LIGHT_M_S / FREQ6, SPEED_OF_LIGHT_M_S / FREQ7,
01386     SPEED_OF_LIGHT_M_S / FREQ8, SPEED_OF_LIGHT_M_S / FREQ9};
01387
01388 const int STRFMT_RTCM2 = 0;   /* stream format: RTCM 2 */
01389 const int STRFMT_RTCM3 = 1;   /* stream format: RTCM 3 */
01390 const int STRFMT_SP3 = 16;    /* stream format: SP3 */
01391 const int STRFMT_RNXCLK = 17; /* stream format: RINEX CLK */
01392 const int STRFMT_SBAS = 18;   /* stream format: SBAS messages */
01393 const int STRFMT_NMEA = 19;   /* stream format: NMEA 0183 */
01394 // const solopt_t solopt_default;   /* default solution output options */
01395
01396 const int MAXSTRRTK = 8; /* max number of stream in RTK server */
01397
01398 /** \} */
01399 /** \} */
01400 #endif  // GNSS_SDR_RTKLIB_H
```

13.163 rtklib_conversions.h File Reference

GNSS-SDR to RTKLIB data structures conversion functions.

```
#include "rtklib.h"
#include <cstdint>
#include <map>
#include <string>
```

Classes

- class HAS_clock_corrections
- class HAS_orbit_corrections
- class HAS_bias_corrections
- class HAS_obs_corrections

Functions

- eph_t **eph_to_rtklib** (const Galileo_Ephemeris &gal_eph)
- eph_t **eph_to_rtklib** (const Galileo_Ephemeris &gal_eph, const std::map< int, HAS_orbit_corrections > &orbit_correction_map, const std::map< int, HAS_clock_corrections > &clock_correction_map)
- eph_t **eph_to_rtklib** (const Gps_Ephemeris &gps_eph, int ref_week=0)
- eph_t **eph_to_rtklib** (const Gps_Ephemeris &gps_eph, const std::map< int, HAS_orbit_corrections > &orbit_correction_map, const std::map< int, HAS_clock_corrections > &clock_correction_map, int ref_↩ week=0)
- eph_t **eph_to_rtklib** (const Gps_CNAV_Ephemeris &gps_cnav_eph)
- eph_t **eph_to_rtklib** (const Beidou_Dnav_Ephemeris &bei_eph)
- eph_t **eph_to_rtklib** (const Beidou_Cnav1_Ephemeris &bei_eph)
- gtime_t glonass_almanac_epoch (const Glonass_Gnav_Almanac &almanac)

Absolute GPST epoch of a GLONASS almanac (N_4/N_A calendar day plus t_lambda_n_A, which is given in MSK = UTC+3).

- alm_t alm_to_rtklib (const Gps_Almanac &gps_alm, int ref_week=0)

  Converts almanac data to RTKLIB's alm_t format.
- alm_t **alm_to_rtklib** (const Galileo_Almanac &gal_alm, int ref_week=0)
- alm_t **alm_to_rtklib** (const Beidou_Dnav_Almanac &bei_alm)
- bool glonass_gnav_is_healthy (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, bool glonass_strict↩ _health=true)

  Broadcast health of a GLONASS satellite according to its GNAV ephemeris.
- geph_t eph_to_rtklib (const Glonass_Gnav_Ephemeris &glonass_gnav_eph, const Glonass_Gnav_Utc_Model &gnav_clock_model, bool glonass_strict_health=true)

  Transforms a Glonass_Gnav_Ephemeris to its RTKLIB counterpart.
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, const std::↩ map< std::string, std::map< int, HAS_obs_corrections > > &has_obs_corr, int week, int band, const HAS_obs_corrections **applied_has_correction, int ref_week)
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, const std::map< std::string, std::map< int, HAS_obs_corrections > > &has_obs_corr, int week, int band, int ref_week=0)
- obsd_t **insert_obs_to_rtklib** (obsd_t &rtklib_obs, const Gnss_Synchro &gnss_synchro, int week, int band, int ref_week=0)

13.163.1 Detailed Description

GNSS-SDR to RTKLIB data structures conversion functions.

Author

2017, Javier Arribas

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file rtklib_conversions.h.

13.164 rtklib_conversions.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_conversions.h
00003  * \brief GNSS-SDR to RTKLIB data structures conversion functions
00004  * \author 2017, Javier Arribas
00005  *
00006  * -------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_RTKLIB_CONVERSIONS_H
00018 #define GNSS_SDR_RTKLIB_CONVERSIONS_H
00019
00020 #include "rtklib.h"
00021 #include <cstdint>
00022 #include <map>
00023 #include <string>
00024
00025 /** \addtogroup PVT
00026  * \{ */
00027 /** \addtogroup RTKLIB_Library
00028  * \{ */
00029
00030
00031 class Beidou_Cnav1_Ephemeris;
00032 class Beidou_Dnav_Almanac;
00033 class Beidou_Dnav_Ephemeris;
00034 class Galileo_Almanac;
00035 class Galileo_Ephemeris;
```

```
00036 class Glonass_Gnav_Almanac;
00037 class Glonass_Gnav_Ephemeris;
00038 class Glonass_Gnav_Utc_Model;
00039 class Gnss_Synchro;
00040 class Gps_Almanac;
00041 class Gps_CNAV_Ephemeris;
00042 class Gps_Ephemeris;
00043
00044 class HAS_clock_corrections
00045 {
00046 public:
00047     HAS_clock_corrections() = default;
00048     float clock_correction_m{};
00049     uint32_t valid_until{};
00050     uint16_t iod{};
00051     uint8_t mask_id{};
00052     uint8_t iod_set_id{};
00053 };
00054
00055 class HAS_orbit_corrections
00056 {
00057 public:
00058     HAS_orbit_corrections() = default;
00059     float radial_m{};
00060     float in_track_m{};
00061     float cross_track_m{};
00062     uint32_t valid_until{};
00063     uint16_t iod{};
00064     uint8_t mask_id{};
00065     uint8_t iod_set_id{};
00066 };
00067
00068 class HAS_bias_corrections
00069 {
00070 public:
00071     HAS_bias_corrections() = default;
00072     float bias{};
00073     uint32_t valid_until{};
00074     uint8_t mask_id{};
00075     uint8_t iod_set_id{};
00076 };
00077
00078 class HAS_obs_corrections
00079 {
00080 public:
00081     HAS_obs_corrections() = default;
00082     std::string signal;
00083     float code_bias_m{};
00084     float phase_bias_cycle{};
00085     uint8_t phase_discontinuity_indicator{};
00086     bool phase_bias_discontinuity{};
00087 };
00088
00089
00090 eph_t eph_to_rtklib(const Galileo_Ephemeris& gal_eph);
00091
00092 eph_t eph_to_rtklib(const Galileo_Ephemeris& gal_eph,
00093     const std::map<int, HAS_orbit_corrections>& orbit_correction_map,
00094     const std::map<int, HAS_clock_corrections>& clock_correction_map);
00095
00096 eph_t eph_to_rtklib(const Gps_Ephemeris& gps_eph,
00097     int ref_week = 0);
00098
00099 eph_t eph_to_rtklib(const Gps_Ephemeris& gps_eph,
00100     const std::map<int, HAS_orbit_corrections>& orbit_correction_map,
00101     const std::map<int, HAS_clock_corrections>& clock_correction_map,
00102     int ref_week = 0);
00103
00104 eph_t eph_to_rtklib(const Gps_CNAV_Ephemeris& gps_cnav_eph);
00105
00106 eph_t eph_to_rtklib(const Beidou_Dnav_Ephemeris& bei_eph);
00107
00108 eph_t eph_to_rtklib(const Beidou_Cnav1_Ephemeris& bei_eph);
00109
00110 /*!
00111  * \brief Absolute GPST epoch of a GLONASS almanac (N_4/N_A calendar day plus
00112  * t_lambda_n_A, which is given in MSK = UTC+3).
00113  * \return Zero gtime_t when the broadcast date is unavailable or invalid.
00114  */
00115 gtime_t glonass_almanac_epoch(const Glonass_Gnav_Almanac& almanac);
00116
00117 /*!
00118  * \brief Converts almanac data to RTKLIB's alm_t format.
00119  *
00120  * \param ref_week Current GPS week, used to resolve the truncated almanac
00121  * week (WNa: 8 bits for GPS, 2 bits for Galileo) to the closest full week so
00122  * that alm.toa is an absolute epoch, as eph_to_rtklib() does for ephemeris.
```

```
00123  * 0 (default) uses WNa as-is; only valid for callers that never propagate
00124  * the orbit (e.g. unit tests checking unrelated fields).
00125  */
00126 alm_t alm_to_rtklib(const Gps_Almanac& gps_alm, int ref_week = 0);
00127 alm_t alm_to_rtklib(const Galileo_Almanac& gal_alm, int ref_week = 0);
00128 alm_t alm_to_rtklib(const Beidou_Dnav_Almanac& bei_alm);
00129
00130 /*!
00131  * \brief Broadcast health of a GLONASS satellite according to its GNAV ephemeris
00132  * \param glonass_gnav_eph GLONASS GNAV Ephemeris structure
00133  * \param glonass_strict_health when true (default), the MSB of the Bn word also marks the
00134  * satellite as unhealthy; when false, only the ln flag is used
00135  * \return true if the satellite does not report a malfunction
00136  */
00137 bool glonass_gnav_is_healthy(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, bool
      glonass_strict_health = true);
00138
00139 /*!
00140  * \brief Transforms a Glonass_Gnav_Ephemeris to its RTKLIB counterpart
00141  * \param glonass_gnav_eph GLONASS GNAV Ephemeris structure
00142  * \param gnav_clock_model GLONASS GNAV UTC model (for the GLONASST to GPST conversion)
00143  * \param glonass_strict_health when true (default), the MSB of the Bn word also marks the
00144  * satellite as unhealthy; when false, only the ln flag is used
00145  * \return Ephemeris structure for RTKLIB parsing
00146  */
00147 geph_t eph_to_rtklib(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, const Glonass_Gnav_Utc_Model&
      gnav_clock_model, bool glonass_strict_health = true);
00148
00149 obsd_t insert_obs_to_rtklib(obsd_t& rtklib_obs,
00150     const Gnss_Synchro& gnss_synchro,
00151     const std::map<std::string, std::map<int, HAS_obs_corrections>>& has_obs_corr,
00152     int week,
00153     int band,
00154     const HAS_obs_corrections** applied_has_correction,
00155     int ref_week);
00156
00157 obsd_t insert_obs_to_rtklib(obsd_t& rtklib_obs,
00158     const Gnss_Synchro& gnss_synchro,
00159     const std::map<std::string, std::map<int, HAS_obs_corrections>>& has_obs_corr,
00160     int week,
00161     int band,
00162     int ref_week = 0);
00163
00164 obsd_t insert_obs_to_rtklib(obsd_t& rtklib_obs, const Gnss_Synchro& gnss_synchro, int week, int band,
      int ref_week = 0);
00165
00166
00167 /** \} */
00168 /** \} */
00169 #endif  // GNSS_SDR_RTKLIB_CONVERSIONS_H
```

13.165 rtklib_ephemeris.h File Reference

satellite ephemeris and clock functions

```
#include "rtklib.h"
```

Functions

- double **var_uraeph** (int ura)
- double **var_sisaeph** (int sisa)
- double **var_urassr** (int ura)
- void **alm2pos** (gtime_t time, const alm_t *alm, double *rs, double *dts)
- double **eph2clk** (gtime_t time, const eph_t *eph)
- void **eph2pos** (gtime_t time, const eph_t *eph, double *rs, double *dts, double *var)
- void **deq** (const double *x, double *xdot, const double *acc)
- void **glorbit** (double t, double *x, const double *acc)
- double **geph2clk** (gtime_t time, const geph_t *geph)
- void **geph2pos** (gtime_t time, const geph_t *geph, double *rs, double *dts, double *var)
- double **seph2clk** (gtime_t time, const seph_t *seph)
- void **seph2pos** (gtime_t time, const seph_t *seph, double *rs, double *dts, double *var)
- eph_t * **seleph** (gtime_t time, int sat, int iode, const nav_t *nav, int bds_eph_sel=-1)
- geph_t * **selgeph** (gtime_t time, int sat, int iode, const nav_t *nav)

- seph_t * **selseph** (gtime_t time, int sat, const nav_t *nav)
- int **ephclk** (gtime_t time, gtime_t teph, int sat, const nav_t *nav, double *dts, int bds_eph_sel=-1)
- int **ephpos** (gtime_t time, gtime_t teph, int sat, const nav_t *nav, int iode, double *rs, double *dts, double *var, int *svh, int bds_eph_sel=-1)
- int **satpos_sbas** (gtime_t time, gtime_t teph, int sat, const nav_t *nav, double *rs, double *dts, double *var, int *svh)
- int **satpos_ssr** (gtime_t time, gtime_t teph, int sat, const nav_t *nav, int opt, double *rs, double *dts, double *var, int *svh)
- int **satpos** (gtime_t time, gtime_t teph, int sat, int ephopt, const nav_t *nav, double *rs, double *dts, double *var, int *svh, int bds_eph_sel=-1)
- void **satposs** (gtime_t teph, const obsd_t *obs, int n, const nav_t *nav, int ephopt, double *rs, double *dts, double *var, int *svh)
- void **satposs_relative** (gtime_t teph, const obsd_t *obs, int n, const nav_t *nav, int ephopt, double *rs, double *dts, double *var, int *svh)

Variables

- const int BDS_EPH_SELECTION_CNAV1_PREFERRED = -2

13.165.1 Detailed Description

satellite ephemeris and clock functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_ephemeris.h.

13.165.2 Variable Documentation

13.165.2.1 BDS_EPH_SELECTION_CNAV1_PREFERRED

```
const int BDS_EPH_SELECTION_CNAV1_PREFERRED = -2
```

Definition at line 57 of file rtklib_ephemeris.h.

13.166 rtklib_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_ephemeris.h
00003  * \brief satellite ephemeris and clock functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
```

```
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * ----------------------------------------------------------------------------
00031  */
00032
00033
00034 #ifndef GNSS_SDR_RTKLIB_EPHEMERIS_H
00035 #define GNSS_SDR_RTKLIB_EPHEMERIS_H
00036
00037 #include "rtklib.h"
00038
00039
00040 double var_uraeph(int ura);
00041 double var_sisaeph(int sisa);
00042 double var_urassr(int ura);
00043 void alm2pos(gtime_t time, const alm_t *alm, double *rs, double *dts);
00044 double eph2clk(gtime_t time, const eph_t *eph);
00045 void eph2pos(gtime_t time, const eph_t *eph, double *rs, double *dts,
00046     double *var);
00047 void deq(const double *x, double *xdot, const double *acc);
00048 void glorbit(double t, double *x, const double *acc);
00049 double geph2clk(gtime_t time, const geph_t *geph);
00050
00051 void geph2pos(gtime_t time, const geph_t *geph, double *rs, double *dts,
00052     double *var);
00053 double seph2clk(gtime_t time, const seph_t *seph);
00054 void seph2pos(gtime_t time, const seph_t *seph, double *rs, double *dts,
00055     double *var);
00056 /* Prefer B-CNAV1 and use DNAV only when no usable B-CNAV1 record exists. */
00057 const int BDS_EPH_SELECTION_CNAV1_PREFERRED = -2;
00058 /* bds_eph_sel: -1=any, 0=DNAV (code not CNAV1/CNAV2),
00059  * BDS_EPH_SELECTION_CNAV1_PREFERRED=B-CNAV1 then DNAV,
00060  * BDS_EPH_SOURCE_CNAV1=B-CNAV1 only,
00061  * BDS_EPH_SOURCE_CNAV2=B-CNAV2 only */
00062 eph_t *seleph(gtime_t time, int sat, int iode, const nav_t *nav, int bds_eph_sel = -1);
00063 geph_t *selgeph(gtime_t time, int sat, int iode, const nav_t *nav);
00064 seph_t *selseph(gtime_t time, int sat, const nav_t *nav);
00065 int ephclk(gtime_t time, gtime_t teph, int sat, const nav_t *nav,
00066     double *dts, int bds_eph_sel = -1);
00067 // satellite position and clock by broadcast ephemeris
00068 int ephpos(gtime_t time, gtime_t teph, int sat, const nav_t *nav,
00069     int iode, double *rs, double *dts, double *var, int *svh, int bds_eph_sel = -1);
00070 int satpos_sbas(gtime_t time, gtime_t teph, int sat, const nav_t *nav,
00071     double *rs, double *dts, double *var, int *svh);
00072 int satpos_ssr(gtime_t time, gtime_t teph, int sat, const nav_t *nav,
00073     int opt, double *rs, double *dts, double *var, int *svh);
00074
00075 int satpos(gtime_t time, gtime_t teph, int sat, int ephopt,
00076     const nav_t *nav, double *rs, double *dts, double *var,
00077     int *svh, int bds_eph_sel = -1);
00078 void satposs(gtime_t teph, const obsd_t *obs, int n, const nav_t *nav,
00079     int ephopt, double *rs, double *dts, double *var, int *svh);
00080 /* Relative same-signal processing may use a DNAV orbit/clock for B1C when
00081  * B-CNAV1 is unavailable; absolute positioning remains CNAV1-only. */
00082 void satposs_relative(gtime_t teph, const obsd_t *obs, int n, const nav_t *nav,
00083     int ephopt, double *rs, double *dts, double *var, int *svh);
00084
00085
00086 #endif  // GNSS_SDR_RTKLIB_EPHEMERIS_H
```

13.167 rtklib_ionex.h File Reference

ionex functions

`#include "rtklib.h"`

Functions

- int **getindex** (double value, const double *range)
- int **nitem** (const double *range)
- int **dataindex** (int i, int j, int k, const int *ndata)
- tec_t * **addtec** (const double *lats, const double *lons, const double *hgts, double rb, nav_t *nav)
- void **readionexdcb** (FILE *fp, double *dcb, double *rms)
- double **readionexh** (FILE *fp, double *lats, double *lons, double *hgts, double *rb, double *nexp, double *dcb, double *rms)
- int **readionexb** (FILE *fp, const double *lats, const double *lons, const double *hgts, double rb, double nexp, nav_t *nav)
- void **combtec** (nav_t *nav)
- void **readtec** (const char *file, nav_t *nav, int opt)
- int **interptec** (const tec_t *tec, int k, const double *posp, double *value, double *rms)
- int **iondelay** (gtime_t time, const tec_t *tec, const double *pos, const double *azel, int opt, double *delay, double *var)
- int **iontec** (gtime_t time, const nav_t *nav, const double *pos, const double *azel, int opt, double *delay, double *var)

Variables

- const double VAR_NOTEC = 30.0 * 30.0
- const double MIN_EL = 0.0
- const double MIN_HGT = -1000.0

13.167.1 Detailed Description

ionex functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References: [1] S.Schear, W.Gurtner and J.Feltens, IONEX: The IONosphere Map EXchange Format Version 1, February 25, 1998 [2] S.Schaer, R.Markus, B.Gerhard and A.S.Timon, Daily Global Ionosphere Maps based on GPS Carrier Phase Data Routinely produced by CODE Analysis Center, Proceeding of the IGS Analysis Center Workshop, 1996
Definition in file rtklib_ionex.h.

13.167.2 Variable Documentation

13.167.2.1 MIN_EL

```
const double MIN_EL = 0.0
```
Definition at line 46 of file rtklib_ionex.h.

13.167.2.2 MIN_HGT

```
const double MIN_HGT = -1000.0
```
Definition at line 47 of file rtklib_ionex.h.

13.167.2.3 VAR_NOTEC

```
const double VAR_NOTEC = 30.0 * 30.0
```

Definition at line 45 of file rtklib_ionex.h.

13.168 rtklib_ionex.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_ionex.h
00003  * \brief ionex functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * References:
00031  *     [1] S.Schear, W.Gurtner and J.Feltens, IONEX: The IONosphere Map EXchange
00032  *         Format Version 1, February 25, 1998
00033  *     [2] S.Schaer, R.Markus, B.Gerhard and A.S.Timon, Daily Global Ionosphere
00034  *         Maps based on GPS Carrier Phase Data Routinely produced by CODE
00035  *         Analysis Center, Proceeding of the IGS Analysis Center Workshop, 1996
00036  *
00037  * ------------------------------------------------------------------------------
00038  */
00039
00040 #ifndef GNSS_SDR_RTKLIB_IONEX_H
00041 #define GNSS_SDR_RTKLIB_IONEX_H
00042
00043 #include "rtklib.h"
00044
00045 const double VAR_NOTEC = 30.0 * 30.0; /* variance of no tec */
00046 const double MIN_EL = 0.0;            /* min elevation angle (rad) */
00047 const double MIN_HGT = -1000.0;       /* min user height (m) */
00048
00049 int getindex(double value, const double *range);
00050
00051 int nitem(const double *range);
00052 int dataindex(int i, int j, int k, const int *ndata);
00053 tec_t *addtec(const double *lats, const double *lons, const double *hgts,
00054     double rb, nav_t *nav);
00055 void readionexdcb(FILE *fp, double *dcb, double *rms);
00056 double readionexh(FILE *fp, double *lats, double *lons, double *hgts,
00057     double *rb, double *nexp, double *dcb, double *rms);
00058 int readionexb(FILE *fp, const double *lats, const double *lons,
00059     const double *hgts, double rb, double nexp, nav_t *nav);
00060 void combtec(nav_t *nav);
00061 void readtec(const char *file, nav_t *nav, int opt);
00062 int interptec(const tec_t *tec, int k, const double *posp, double *value,
00063     double *rms);
00064
00065 int iondelay(gtime_t time, const tec_t *tec, const double *pos,
00066     const double *azel, int opt, double *delay, double *var);
00067 int iontec(gtime_t time, const nav_t *nav, const double *pos,
00068     const double *azel, int opt, double *delay, double *var);
00069
00070 #endif  // GNSS_SDR_RTKLIB_IONEX_H
```

13.169 rtklib_lambda.h File Reference

Integer ambiguity resolution.
`#include "rtklib.h"`

Macros

- #define SGN_LAMBDA(x)
- #define ROUND_LAMBDA(x)
- #define SWAP_LAMBDA(x, y)

Functions

- int **LD** (int n, const double *Q, double *L, double *D)
- void **gauss** (int n, double *L, double *Z, int i, int j)
- void **perm** (int n, double *L, double *D, int j, double del, double *Z)
- void **reduction** (int n, double *L, double *D, double *Z)
- int **search** (int n, int m, const double *L, const double *D, const double *zs, double *zn, double *s)
- int **lambda** (int n, int m, const double *a, const double *Q, double *F, double *s)
- int **lambda_reduction** (int n, const double *Q, double *Z)
- int **lambda_search** (int n, int m, const double *a, const double *Q, double *F, double *s)

Variables

- const int LOOPMAX = 10000

13.169.1 Detailed Description

Integer ambiguity resolution.

Authors

- 2007-2008, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2008, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References: [1] P.J.G.Teunissen, The least-square ambiguity decorrelation adjustment: a method for fast GPS ambiguity estimation, J.Geodesy, Vol.70, 65-82, 1995 [2] X.-W.Chang, X.Yang, T.Zhou, MLAMBDA: A modified LAMBDA method for integer least-squares estimation, J.Geodesy, Vol.79, 552-565, 2005
Definition in file rtklib_lambda.h.

13.169.2 Macro Definition Documentation

13.169.2.1 ROUND_LAMBDA

```
#define ROUND_LAMBDA(
                x)
```

Value:

```
(floor((x) + 0.5))
```

Definition at line 49 of file rtklib_lambda.h.

13.169.2.2 SGN_LAMBDA

```
#define SGN_LAMBDA(
                x)
```

Value:

```
((x) <= 0.0 ? -1.0 : 1.0)
```

Definition at line 48 of file rtklib_lambda.h.

13.169.2.3 SWAP_LAMBDA

```
#define SWAP_LAMBDA(
                x,
                y)
```

Value:

```
do                      \
    {                   \
        double tmp_;    \
        tmp_ = x;       \
        x = y;          \
        y = tmp_;       \
    }                   \
while (0)
```

Definition at line 50 of file rtklib_lambda.h.

13.169.3 Variable Documentation

13.169.3.1 LOOPMAX

```
const int LOOPMAX = 10000
```

Definition at line 47 of file rtklib_lambda.h.

13.170 rtklib_lambda.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_lambda.h
00003  * \brief Integer ambiguity resolution
00004  * \authors <ul>
00005  *          <li> 2007-2008, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2008, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * References:
00031  *     [1] P.J.G.Teunissen, The least-square ambiguity decorrelation adjustment:
00032  *         a method for fast GPS ambiguity estimation, J.Geodesy, Vol.70, 65-82,
00033  *         1995
00034  *     [2] X.-W.Chang, X.Yang, T.Zhou, MLAMBDA: A modified LAMBDA method for
00035  *         integer least-squares estimation, J.Geodesy, Vol.79, 552-565, 2005
00036  *
00037  * ------------------------------------------------------------------------------
00038  */
00039
00040 #ifndef GNSS_SDR_RTKLIB_LAMBDA_H
```

```
00041 #define GNSS_SDR_RTKLIB_LAMBDA_H
00042
00043
00044 #include "rtklib.h"
00045
00046 /* constants/macros ----------------------------------------------------------*/
00047 const int LOOPMAX = 10000; /* maximum count of search loop */
00048 #define SGN_LAMBDA(x) ((x) <= 0.0 ? -1.0 : 1.0)
00049 #define ROUND_LAMBDA(x) (floor((x) + 0.5))
00050 #define SWAP_LAMBDA(x, y) \
00051     do                    \
00052         {                 \
00053             double tmp_;  \
00054             tmp_ = x;     \
00055             x = y;        \
00056             y = tmp_;     \
00057         }                 \
00058     while (0)
00059
00060 int LD(int n, const double *Q, double *L, double *D);
00061 void gauss(int n, double *L, double *Z, int i, int j);
00062 void perm(int n, double *L, double *D, int j, double del, double *Z);
00063 void reduction(int n, double *L, double *D, double *Z);
00064 int search(int n, int m, const double *L, const double *D,
00065     const double *zs, double *zn, double *s);
00066
00067 int lambda(int n, int m, const double *a, const double *Q, double *F, double *s);
00068
00069 int lambda_reduction(int n, const double *Q, double *Z);
00070
00071 int lambda_search(int n, int m, const double *a, const double *Q,
00072     double *F, double *s);
00073
00074
00075 #endif
```

13.171 rtklib_pntpos.h File Reference

standard code-based positioning

```
#include "rtklib.h"
#include "rtklib_rtkcmn.h"
```

Functions

- double **varerr** (const prcopt_t *opt, const obsd_t *obs, double el, int sys)
- double **gettgd** (int sat, const nav_t *nav)
- double **gettgd** (int sat, const nav_t *nav, int tgd_index)
- double **gettgd_bds_by_obs_code** (int sat, const nav_t *nav, unsigned char obs_code)
- int **galileo_bgd_index** (unsigned char observation_code, int sat, const nav_t *nav)
- double **getiscl1** (int sat, const nav_t *nav)
- double **getiscl2** (int sat, const nav_t *nav)
- double **getiscl5i** (int sat, const nav_t *nav)
- double **getiscl5q** (int sat, const nav_t *nav)
- double **prange** (const obsd_t *obs, const nav_t *nav, const double *azel, int iter, const prcopt_t *opt, double *var, double *iono_scale)
- int **ionocorr** (gtime_t time, const nav_t *nav, int sat, const double *pos, const double *azel, int ionoopt, double *ion, double *var, unsigned char obs_code=0)
- int **tropcorr** (gtime_t time, const nav_t *nav, const double *pos, const double *azel, int tropopt, double *trp, double *var)
- int **rescode** (int iter, const obsd_t *obs, int n, const double *rs, const double *dts, const double *vare, const int *svh, const nav_t *nav, const double *x, const prcopt_t *opt, double *v, double *H, double *var, double *azel, int *vsat, double *resp, int *ns)
- int **valsol** (const double *azel, const int *vsat, int n, const prcopt_t *opt, const double *v, int nv, int nx, char *msg)
- int **estpos** (const obsd_t *obs, int n, const double *rs, const double *dts, const double *vare, const int *svh, const nav_t *nav, const prcopt_t *opt, sol_t *sol, double *azel, int *vsat, double *resp, char *msg)

- int **raim_fde** (const obsd_t *obs, int n, const double *rs, const double *dts, const double *vare, const int *svh, const nav_t *nav, const prcopt_t *opt, sol_t *sol, double *azel, int *vsat, double *resp, char *msg)
- int **resdop** (const obsd_t *obs, int n, const double *rs, const double *dts, const nav_t *nav, const double *rr, const double *x, const double *azel, const int *vsat, double *v, double *H)
- void **estvel** (const obsd_t *obs, int n, const double *rs, const double *dts, const nav_t *nav, const prcopt_t *opt, sol_t *sol, const double *azel, const int *vsat)
- int pntpos (const obsd_t *obs, int n, const nav_t *nav, const prcopt_t *opt, sol_t *sol, double *azel, ssat_t *ssat, char *msg)

  *single-point positioning compute receiver position, velocity, clock bias by single-point positioning with pseudorange and doppler observables args : obsd_t *obs I observation data int n I number of observation data nav_t *nav I navigation data prcopt_t *opt I processing options sol_t *sol IO solution double *azel IO azimuth/elevation angle (rad) (NULL: no output) ssat_t *ssat IO satellite status (NULL: no output) char *msg O error message for error exit return : status(1:ok,0:error) notes : assuming sbas-gps, galileo-gps, qzss-gps, compass-gps time offset and receiver bias are negligible (only involving glonass-gps time offset and receiver bias)*

Variables

- const int NX = 4 + 4
- const int MAXITR = 10

  max number of iteration for point pos
- const double VARERR_MIN_EL = 5.0 * D2R

  minimum elevation for measurement error variance (rad)
- const double ERR_ION = 5.0

  ionospheric delay std (m)
- const double ERR_TROP = 3.0

  tropspheric delay std (m)

13.171.1 Detailed Description

standard code-based positioning

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_pntpos.h.

13.171.2 Function Documentation

13.171.2.1 pntpos()

```
int pntpos (
            const obsd_t * obs,
            int n,
            const nav_t * nav,
            const prcopt_t * opt,
            sol_t * sol,
            double * azel,
```

```
            ssat_t * ssat,
            char * msg)
```

single-point positioning compute receiver position, velocity, clock bias by single-point positioning with pseudorange and doppler observables args : obsd_t *obs I observation data int n I number of observation data nav_t *nav I navigation data prcopt_t *opt I processing options sol_t *sol IO solution double *azel IO azimuth/elevation angle (rad) (NULL: no output) ssat_t *ssat IO satellite status (NULL: no output) char *msg O error message for error exit return : status(1:ok,0:error) notes : assuming sbas-gps, galileo-gps, qzss-gps, compass-gps time offset and receiver bias are negligible (only involving glonass-gps time offset and receiver bias)
References pntpos().
Referenced by pntpos().

13.171.3 Variable Documentation

13.171.3.1 ERR_ION

```
const double ERR_ION = 5.0
```

ionospheric delay std (m)
Definition at line 43 of file rtklib_pntpos.h.

13.171.3.2 ERR_TROP

```
const double ERR_TROP = 3.0
```

tropspheric delay std (m)
Definition at line 44 of file rtklib_pntpos.h.

13.171.3.3 MAXITR

```
const int MAXITR = 10
```

max number of iteration for point pos
Definition at line 41 of file rtklib_pntpos.h.

13.171.3.4 NX

```
const int NX = 4 + 4
```

13.171.4 of estimated parameters: pos (3), GPS clock, GLO/GAL/BDS/QZS-GPS offsets

Definition at line 40 of file rtklib_pntpos.h.

13.171.4.1 VARERR_MIN_EL

```
const double VARERR_MIN_EL = 5.0 * D2R
```

minimum elevation for measurement error variance (rad)
Definition at line 42 of file rtklib_pntpos.h.

13.172 rtklib_pntpos.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_pntpos.h
00003  * \brief standard code-based positioning
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
```

```
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * -----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * -----------------------------------------------------------------------------
00031  */
00032
00033 #ifndef GNSS_SDR_RTKLIB_PNTPOS_H
00034 #define GNSS_SDR_RTKLIB_PNTPOS_H
00035
00036 #include "rtklib.h"
00037 #include "rtklib_rtkcmn.h"
00038
00039 /* constants ---------------------------------------------------------------*/
00040 const int NX = 4 + 4;                    //!< # of estimated parameters: pos (3), GPS clock,
      GLO/GAL/BDS/QZS-GPS offsets
00041 const int MAXITR = 10;                   //!< max number of iteration for point pos
00042 const double VARERR_MIN_EL = 5.0 * D2R;  //!< minimum elevation for measurement error variance (rad)
00043 const double ERR_ION = 5.0;              //!< ionospheric delay std (m)
00044 const double ERR_TROP = 3.0;             //!< tropspheric delay std (m)
00045
00046
00047 /* pseudorange measurement error variance ----------------------------------*/
00048 double varerr(const prcopt_t *opt, const obsd_t *obs, double el, int sys);
00049
00050 /* get tgd parameter (m) ---------------------------------------------------*/
00051 double gettgd(int sat, const nav_t *nav);
00052 double gettgd(int sat, const nav_t *nav, int tgd_index);
00053 /* BDS DNAV/CNAV1 TGD selection by observation code (CODE_L1D/L1P/L1X/...) */
00054 double gettgd_bds_by_obs_code(int sat, const nav_t *nav, unsigned char obs_code);
00055
00056 /* select Galileo BGD from observation and ephemeris provenance ------------*/
00057 int galileo_bgd_index(unsigned char observation_code, int sat, const nav_t *nav);
00058
00059 /* get isc parameter (m) ---------------------------------------------------*/
00060 double getiscl1(int sat, const nav_t *nav);
00061 double getiscl2(int sat, const nav_t *nav);
00062 double getiscl5i(int sat, const nav_t *nav);
00063 double getiscl5q(int sat, const nav_t *nav);
00064
00065 /* psendorange with code bias correction -----------------------------------
00066  * iono_scale (O) is the multiplier the caller must apply to the modeled L1
00067  * ionospheric delay so that it is consistent with the returned pseudorange:
00068  * 1.0 for L1-referenced measurements, (f_L1/f_band)^2 for single-band
00069  * measurements on another band, 0.0 for ionosphere-free combinations  */
00070 double prange(const obsd_t *obs, const nav_t *nav, const double *azel,
00071     int iter, const prcopt_t *opt, double *var, double *iono_scale);
00072
00073 /* ionospheric correction --------------------------------------------------
00074  * compute ionospheric correction
00075  * args   : gtime_t time     I   time
00076  *          nav_t  *nav      I   navigation data
00077  *          int    sat       I   satellite number
00078  *          double *pos      I   receiver position {lat,lon,h} (rad|m)
00079  *          double *azel     I   azimuth/elevation angle {az,el} (rad)
00080  *          int    ionoopt   I   ionospheric correction option (IONOOPT_???)
00081  *          double *ion      O   ionospheric delay (L1) (m)
00082  *          double *var      O   ionospheric delay (L1) variance (m^2)
00083  *          unsigned char obs_code I optional observation code (CODE_???); for BDS
00084  *                                 CODE_L1D/L1P/L1X selects BDGIM at FREQ1 (B1C)
00085  * return : status(1:ok,0:error)
00086  *-----------------------------------------------------------------------------*/
00087 int ionocorr(gtime_t time, const nav_t *nav, int sat, const double *pos,
00088     const double *azel, int ionoopt, double *ion, double *var, unsigned char obs_code = 0);
00089 /* tropospheric correction -------------------------------------------------
00090  * compute tropospheric correction
00091  * args   : gtime_t time     I   time
00092  *          nav_t  *nav      I   navigation data
00093  *          double *pos      I   receiver position {lat,lon,h} (rad|m)
00094  *          double *azel     I   azimuth/elevation angle {az,el} (rad)
00095  *          int    tropopt   I   tropospheric correction option (TROPOPT_???)
00096  *          double *trp      O   tropospheric delay (m)
00097  *          double *var      O   tropospheric delay variance (m^2)
00098  * return : status(1:ok,0:error)
00099  *-----------------------------------------------------------------------------*/
00100 int tropcorr(gtime_t time, const nav_t *nav, const double *pos,
00101     const double *azel, int tropopt, double *trp, double *var);
```

```
00102
00103 /* pseudorange residuals ------------------------------------------------------*/
00104 int rescode(int iter, const obsd_t *obs, int n, const double *rs,
00105     const double *dts, const double *vare, const int *svh,
00106     const nav_t *nav, const double *x, const prcopt_t *opt,
00107     double *v, double *H, double *var, double *azel, int *vsat,
00108     double *resp, int *ns);
00109
00110 /* validate solution ---------------------------------------------------------*/
00111 int valsol(const double *azel, const int *vsat, int n,
00112     const prcopt_t *opt, const double *v, int nv, int nx,
00113     char *msg);
00114
00115 /* estimate receiver position ------------------------------------------------*/
00116 int estpos(const obsd_t *obs, int n, const double *rs, const double *dts,
00117     const double *vare, const int *svh, const nav_t *nav,
00118     const prcopt_t *opt, sol_t *sol, double *azel, int *vsat,
00119     double *resp, char *msg);
00120
00121 /* raim fde (failure detection and exclution) --------------------------------*/
00122 int raim_fde(const obsd_t *obs, int n, const double *rs,
00123     const double *dts, const double *vare, const int *svh,
00124     const nav_t *nav, const prcopt_t *opt, sol_t *sol,
00125     double *azel, int *vsat, double *resp, char *msg);
00126
00127 /* doppler residuals ---------------------------------------------------------*/
00128 int resdop(const obsd_t *obs, int n, const double *rs, const double *dts,
00129     const nav_t *nav, const double *rr, const double *x,
00130     const double *azel, const int *vsat, double *v, double *H);
00131
00132 /* estimate receiver velocity ------------------------------------------------*/
00133 void estvel(const obsd_t *obs, int n, const double *rs, const double *dts,
00134     const nav_t *nav, const prcopt_t *opt, sol_t *sol,
00135     const double *azel, const int *vsat);
00136
00137 /*!
00138  * \brief single-point positioning
00139  * compute receiver position, velocity, clock bias by single-point positioning
00140  * with pseudorange and doppler observables
00141  * args   : obsd_t *obs      I   observation data
00142  *          int    n         I   number of observation data
00143  *          nav_t  *nav      I   navigation data
00144  *          prcopt_t *opt    I   processing options
00145  *          sol_t  *sol      IO  solution
00146  *          double *azel     IO  azimuth/elevation angle (rad) (NULL: no output)
00147  *          ssat_t *ssat     IO  satellite status              (NULL: no output)
00148  *          char   *msg      O   error message for error exit
00149  * return : status(1:ok,0:error)
00150  * notes  : assuming sbas-gps, galileo-gps, qzss-gps, compass-gps time offset and
00151  *          receiver bias are negligible (only involving glonass-gps time offset
00152  *          and receiver bias)
00153  */
00154 int pntpos(const obsd_t *obs, int n, const nav_t *nav,
00155     const prcopt_t *opt, sol_t *sol, double *azel, ssat_t *ssat,
00156     char *msg);
00157
00158 #endif  // GNSS_SDR_RTKLIB_PNTPOS_H
```

13.173 rtklib_ppp.h File Reference

Precise Point Positioning.

`#include "rtklib.h"`

Macros

- #define MIN_PPP(x, y)
- #define ROUND_PPP(x)
- #define SWAP_I(x, y)
- #define SWAP_D(x, y)

Functions

- double **lam_LC** (int i, int j, int k)
- double **L_LC** (int i, int j, int k, const double *L)
- double **P_LC** (int i, int j, int k, const double *P)

- double **var_LC** (int i, int j, int k, double sig)
- double **q_gamma** (double a, double x, double log_gamma_a)
- double **p_gamma** (double a, double x, double log_gamma_a)
- double **f_erfc** (double x)
- double **conffunc** (int N, double B, double sig)
- void **average_LC** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav, const double *azel)
- int **fix_amb_WL** (rtk_t *rtk, const nav_t *nav, int sat1, int sat2, int *NW)
- int **is_depend** (int sat1, int sat2, int *flgs, int *max_flg)
- int **sel_amb** (int *sat1, int *sat2, double *N, double *var, int n)
- int **fix_sol** (rtk_t *rtk, const int *sat1, const int *sat2, const double *NC, int n)
- int **fix_amb_ROUND** (rtk_t *rtk, int *sat1, int *sat2, const int *NW, int n)
- int **fix_amb_ILS** (rtk_t *rtk, int *sat1, int *sat2, int *NW, int n)
- int **pppamb** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav, const double *azel)
- void **pppoutsolstat** (rtk_t *rtk, int level, FILE *fp)
- void **testeclipse** (const obsd_t *obs, int n, const nav_t *nav, double *rs)
- double **varerr** (int sat, int sys, double el, int type, const prcopt_t *opt)
- void **initx** (rtk_t *rtk, double xi, double var, int i)
- int **ifmeas** (const obsd_t *obs, const nav_t *nav, const double *azel, const prcopt_t *opt, const double *dantr, const double *dants, double phw, double *meas, double *var)
- double **gettgd_ppp** (int sat, const nav_t *nav)
- double **gettgd_ppp** (int sat, const nav_t *nav, int tgd_index)
- int **corr_ion** (gtime_t time, const nav_t *nav, int sat, const double *pos, const double *azel, int ionoopt, double *ion, double *var, int *brk)
- int **corrmeas** (const obsd_t *obs, const nav_t *nav, const double *pos, const double *azel, const prcopt_t *opt, const double *dantr, const double *dants, double phw, double *meas, double *var, int *brk)
- double **gfmeas** (const obsd_t *obs, const nav_t *nav)
- void **udpos_ppp** (rtk_t *rtk)
- void **udclk_ppp** (rtk_t *rtk)
- void **udtrop_ppp** (rtk_t *rtk)
- void **detslp_ll** (rtk_t *rtk, const obsd_t *obs, int n)
- void **detslp_gf** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)
- void **udbias_ppp** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)
- void **udstate_ppp** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)
- void **satantpcv** (const double *rs, const double *rr, const pcv_t *pcv, double *dant)
- double **prectrop** (gtime_t time, const double *pos, const double *azel, const prcopt_t *opt, const double *x, double *dtdx, double *var)
- int **res_ppp** (int iter, const obsd_t *obs, int n, const double *rs, const double *dts, const double *vare, const int *svh, const nav_t *nav, const double *x, rtk_t *rtk, double *v, double *H, double *R, double *azel)
- int **pppnx** (const prcopt_t *opt)
- void **pppos** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)

Variables

- const double MIN_ARC_GAP = 300.0
- const double CONST_AMB = 0.001
- const double THRES_RES = 0.3
- const double LOG_PI = 1.14472988584940017
- const double SQRT2 = 1.41421356237309510
- const double VAR_POS_PPP = std::pow(100.0, 2.0)
- const double VAR_CLK = std::pow(100.0, 2.0)
- const double VAR_ZTD = std::pow(0.3, 2.0)
- const double VAR_GRA_PPP = std::pow(0.001, 2.0)
- const double VAR_BIAS = std::pow(100.0, 2.0)
- const double VAR_IONO_OFF = std::pow(10.0, 2.0)

13.173.1 Detailed Description

Precise Point Positioning.

Authors

- 2007-2008, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2008, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_ppp.h.

13.173.2 Macro Definition Documentation

13.173.2.1 MIN_PPP

```
#define MIN_PPP(
                x,
                y)
```

Value:

```
((x) <= (y) ? (x) : (y))
```

Definition at line 40 of file rtklib_ppp.h.

13.173.2.2 ROUND_PPP

```
#define ROUND_PPP(
                x)
```

Value:

```
static_cast<int>(floor((x) + 0.5))
```

Definition at line 41 of file rtklib_ppp.h.

13.173.2.3 SWAP_D

```
#define SWAP_D(
                x,
                y)
```

Value:

```
do                      \
    {                   \
        double _z = x;  \
        x = y;          \
        y = _z;         \
    }                   \
while (0)
```

Definition at line 51 of file rtklib_ppp.h.

13.173.2.4 SWAP_I

```
#define SWAP_I(
                x,
                y)
```

Value:

```
do                  \
    {               \
```

```
        int _z = x; \
        x = y;      \
        y = _z;     \
    }               \
while (0)
```

Definition at line 43 of file rtklib_ppp.h.

13.173.3 Variable Documentation

13.173.3.1 CONST_AMB

```
const double CONST_AMB = 0.001
```

Definition at line 61 of file rtklib_ppp.h.

13.173.3.2 LOG_PI

```
const double LOG_PI = 1.14472988584940017
```

Definition at line 63 of file rtklib_ppp.h.

13.173.3.3 MIN_ARC_GAP

```
const double MIN_ARC_GAP = 300.0
```

Definition at line 60 of file rtklib_ppp.h.

13.173.3.4 SQRT2

```
const double SQRT2 = 1.41421356237309510
```

Definition at line 64 of file rtklib_ppp.h.

13.173.3.5 THRES_RES

```
const double THRES_RES = 0.3
```

Definition at line 62 of file rtklib_ppp.h.

13.173.3.6 VAR_BIAS

```
const double VAR_BIAS = std::pow(100.0, 2.0)
```

Definition at line 70 of file rtklib_ppp.h.

13.173.3.7 VAR_CLK

```
const double VAR_CLK = std::pow(100.0, 2.0)
```

Definition at line 67 of file rtklib_ppp.h.

13.173.3.8 VAR_GRA_PPP

```
const double VAR_GRA_PPP = std::pow(0.001, 2.0)
```

Definition at line 69 of file rtklib_ppp.h.

13.173.3.9 VAR_IONO_OFF

```
const double VAR_IONO_OFF = std::pow(10.0, 2.0)
```

Definition at line 72 of file rtklib_ppp.h.

13.173.3.10 VAR_POS_PPP

```
const double VAR_POS_PPP = std::pow(100.0, 2.0)
```

Definition at line 66 of file rtklib_ppp.h.

13.173.3.11 VAR_ZTD

```
const double VAR_ZTD = std::pow(0.3, 2.0)
```

Definition at line 68 of file rtklib_ppp.h.

13.174 rtklib_ppp.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_ppp.h
00003  * \brief Precise Point Positioning
00004  * \authors <ul>
00005  *          <li> 2007-2008, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2008, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * ------------------------------------------------------------------------------
00031  */
00032
00033
00034 #ifndef GNSS_SDR_RTKLIB_PPP_H
00035 #define GNSS_SDR_RTKLIB_PPP_H
00036
00037 #include "rtklib.h"
00038
00039
00040 #define MIN_PPP(x, y) ((x) <= (y) ? (x) : (y))
00041 #define ROUND_PPP(x) static_cast<int>(floor((x) + 0.5))
00042
00043 #define SWAP_I(x, y)     \
00044     do                   \
00045         {                \
00046             int _z = x; \
00047             x = y;       \
00048             y = _z;      \
00049         }                \
00050     while (0)
00051 #define SWAP_D(x, y)        \
00052     do                      \
00053         {                   \
00054             double _z = x; \
00055             x = y;          \
00056             y = _z;         \
00057         }                   \
00058     while (0)
00059
00060 const double MIN_ARC_GAP = 300.0;          /* min arc gap (s) */
00061 const double CONST_AMB = 0.001;            /* constraint to fixed ambiguity */
00062 const double THRES_RES = 0.3;              /* threshold of residuals test (m) */
00063 const double LOG_PI = 1.14472988584940017; /* log(pi) */
00064 const double SQRT2 = 1.41421356237309510;  /* sqrt(2) */
00065
00066 const double VAR_POS_PPP = std::pow(100.0, 2.0); /* init variance receiver position (m^2) */
00067 const double VAR_CLK = std::pow(100.0, 2.0);     /* init variance receiver clock (m^2) */
00068 const double VAR_ZTD = std::pow(0.3, 2.0);       /* init variance ztd (m^2) */
00069 const double VAR_GRA_PPP = std::pow(0.001, 2.0); /* init variance gradient (m^2) */
00070 const double VAR_BIAS = std::pow(100.0, 2.0);    /* init variance phase-bias (m^2) */
00071
00072 const double VAR_IONO_OFF = std::pow(10.0, 2.0); /* variance of iono-model-off */
00073
00074
00075 /* functions originally included in RTKLIB/src/ppp_ar.c v2.4.2*/
```

```
00076 double lam_LC(int i, int j, int k);
00077
00078 double L_LC(int i, int j, int k, const double *L);
00079
00080 double P_LC(int i, int j, int k, const double *P);
00081
00082 double var_LC(int i, int j, int k, double sig);
00083
00084 double q_gamma(double a, double x, double log_gamma_a);
00085
00086 double p_gamma(double a, double x, double log_gamma_a);
00087
00088 double f_erfc(double x);
00089
00090 double conffunc(int N, double B, double sig);
00091
00092 void average_LC(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav, const double *azel);
00093
00094 int fix_amb_WL(rtk_t *rtk, const nav_t *nav, int sat1, int sat2, int *NW);
00095
00096 int is_depend(int sat1, int sat2, int *flgs, int *max_flg);
00097
00098 int sel_amb(int *sat1, int *sat2, double *N, double *var, int n);
00099
00100 int fix_sol(rtk_t *rtk, const int *sat1, const int *sat2, const double *NC, int n);
00101
00102 int fix_amb_ROUND(rtk_t *rtk, int *sat1, int *sat2, const int *NW, int n);
00103
00104 int fix_amb_ILS(rtk_t *rtk, int *sat1, int *sat2, int *NW, int n);
00105
00106 int pppamb(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav, const double *azel);
00107
00108
00109 /* functions originally included in RTKLIB/src/ppp.c v2.4.2 */
00110 void pppoutsolstat(rtk_t *rtk, int level, FILE *fp);
00111
00112 void testeclipse(const obsd_t *obs, int n, const nav_t *nav, double *rs);
00113
00114 double varerr(int sat, int sys, double el, int type, const prcopt_t *opt);
00115
00116 void initx(rtk_t *rtk, double xi, double var, int i);
00117
00118 int ifmeas(const obsd_t *obs, const nav_t *nav, const double *azel,
00119     const prcopt_t *opt, const double *dantr, const double *dants,
00120     double phw, double *meas, double *var);
00121
00122 double gettgd_ppp(int sat, const nav_t *nav);
00123 double gettgd_ppp(int sat, const nav_t *nav, int tgd_index);
00124
00125 int corr_ion(gtime_t time, const nav_t *nav, int sat, const double *pos,
00126     const double *azel, int ionoopt, double *ion, double *var,
00127     int *brk);
00128
00129 int corrmeas(const obsd_t *obs, const nav_t *nav, const double *pos,
00130     const double *azel, const prcopt_t *opt,
00131     const double *dantr, const double *dants, double phw,
00132     double *meas, double *var, int *brk);
00133
00134 double gfmeas(const obsd_t *obs, const nav_t *nav);
00135
00136 void udpos_ppp(rtk_t *rtk);
00137
00138 void udclk_ppp(rtk_t *rtk);
00139
00140 void udtrop_ppp(rtk_t *rtk);
00141
00142 void detslp_ll(rtk_t *rtk, const obsd_t *obs, int n);
00143
00144 void detslp_gf(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav);
00145
00146 void udbias_ppp(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav);
00147
00148 void udstate_ppp(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav);
00149
00150 void satantpcv(const double *rs, const double *rr, const pcv_t *pcv, double *dant);
00151
00152 double prectrop(gtime_t time, const double *pos, const double *azel,
00153     const prcopt_t *opt, const double *x, double *dtdx,
00154     double *var);
00155
00156 int res_ppp(int iter, const obsd_t *obs, int n, const double *rs,
00157     const double *dts, const double *vare, const int *svh,
00158     const nav_t *nav, const double *x, rtk_t *rtk, double *v,
00159     double *H, double *R, double *azel);
00160
00161 int pppnx(const prcopt_t *opt);
00162
```

```
00163 void pppos(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav);
00164
00165 #endif
```

13.175 rtklib_preceph.h File Reference

precise ephemeris and clock functions
`#include "rtklib.h"`

Functions

- int **code2sys** (char code)
- int **readsp3h** (FILE *fp, gtime_t *time, char *type, int *sats, double *bfact, char *tsys)
- int **addpeph** (nav_t *nav, peph_t *peph)
- void **readsp3b** (FILE *fp, char type, int *sats, int ns, const double *bfact, char *tsys, int index, int opt, nav_t *nav)
- int **cmppeph** (const void *p1, const void *p2)
- void **combpeph** (nav_t *nav, int opt)
- void **readsp3** (const char *file, nav_t *nav, int opt)
- int **readsap** (const char *file, gtime_t time, nav_t *nav)
- int **readdcbf** (const char *file, nav_t *nav, const sta_t *sta)
- int **readdcb** (const char *file, nav_t *nav, const sta_t *sta)
- double **interppol** (const double *x, double *y, int n)
- int **pephpos** (gtime_t time, int sat, const nav_t *nav, double *rs, double *dts, double *vare, double *varc)
- int **pephclk** (gtime_t time, int sat, const nav_t *nav, double *dts, double *varc)
- void **satantoff** (gtime_t time, const double *rs, int sat, const nav_t *nav, double *dant)
- int **peph2pos** (gtime_t time, int sat, const nav_t *nav, int opt, double *rs, double *dts, double *var)

Variables

- const int NMAX = 10
- const double MAXDTE = 900.0
- const double EXTERR_CLK = 1e-3
- const double EXTERR_EPH = 5e-7

13.175.1 Detailed Description

precise ephemeris and clock functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References : [1] S.Hilla, The Extended Standard Product 3 Orbit Format (SP3-c), 12 February, 2007 [2] J.Ray, W. Gurtner, RINEX Extensions to Handle Clock Information, 27 August, 1998 [3] D.D.McCarthy, IERS Technical Note 21, IERS Conventions 1996, July 1996 [4] D.A.Vallado, Fundamentals of Astrodynamics and Applications 2nd ed, Space Technology Library, 2004
Definition in file rtklib_preceph.h.

13.175.2 Variable Documentation

13.175.2.1 EXTERR_CLK

```
const double EXTERR_CLK = 1e-3
```
Definition at line 51 of file rtklib_preceph.h.

13.175.2.2 EXTERR_EPH

```
const double EXTERR_EPH = 5e-7
```
Definition at line 52 of file rtklib_preceph.h.

13.175.2.3 MAXDTE

```
const double MAXDTE = 900.0
```
Definition at line 50 of file rtklib_preceph.h.

13.175.2.4 NMAX

```
const int NMAX = 10
```
Definition at line 49 of file rtklib_preceph.h.

13.176 rtklib_preceph.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_preceph.h
00003  * \brief precise ephemeris and clock functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * References :
00032  *    [1] S.Hilla, The Extended Standard Product 3 Orbit Format (SP3-c),
00033  *        12 February, 2007
00034  *    [2] J.Ray, W.Gurtner, RINEX Extensions to Handle Clock Information,
00035  *        27 August, 1998
00036  *    [3] D.D.McCarthy, IERS Technical Note 21, IERS Conventions 1996, July 1996
00037  *    [4] D.A.Vallado, Fundamentals of Astrodynamics and Applications 2nd ed,
00038  *        Space Technology Library, 2004
00039  *
00040  * ------------------------------------------------------------------------------
00041  */
00042
00043 #ifndef GNSS_SDR_RTKLIB_PRECEPH_H
00044 #define GNSS_SDR_RTKLIB_PRECEPH_H
00045
00046 #include "rtklib.h"
00047
00048
00049 const int NMAX = 10;            /* order of polynomial interpolation */
00050 const double MAXDTE = 900.0;    /* max time difference to ephem time (s) */
```

```
00051 const double EXTERR_CLK = 1e-3; /* extrapolation error for clock (m/s) */
00052 const double EXTERR_EPH = 5e-7; /* extrapolation error for ephem (m/s^2) */
00053
00054 int code2sys(char code);
00055
00056 int readsp3h(FILE *fp, gtime_t *time, char *type, int *sats,
00057     double *bfact, char *tsys);
00058
00059 int addpeph(nav_t *nav, peph_t *peph);
00060
00061 void readsp3b(FILE *fp, char type, int *sats, int ns, const double *bfact,
00062     char *tsys, int index, int opt, nav_t *nav);
00063
00064 int cmppeph(const void *p1, const void *p2);
00065
00066 void combpeph(nav_t *nav, int opt);
00067
00068 void readsp3(const char *file, nav_t *nav, int opt);
00069
00070 int readsap(const char *file, gtime_t time, nav_t *nav);
00071
00072 int readdcbf(const char *file, nav_t *nav, const sta_t *sta);
00073
00074 int readdcb(const char *file, nav_t *nav, const sta_t *sta);
00075
00076 double interppol(const double *x, double *y, int n);
00077
00078 int pephpos(gtime_t time, int sat, const nav_t *nav, double *rs,
00079     double *dts, double *vare, double *varc);
00080
00081 int pephclk(gtime_t time, int sat, const nav_t *nav, double *dts,
00082     double *varc);
00083
00084 void satantoff(gtime_t time, const double *rs, int sat, const nav_t *nav,
00085     double *dant);
00086
00087 int peph2pos(gtime_t time, int sat, const nav_t *nav, int opt,
00088     double *rs, double *dts, double *var);
00089
00090 #endif  // GNSS_SDR_RTKLIB_PRECEPH_H
```

13.177 rtklib_rtcm.h File Reference

RTCM functions headers.

```
#include "rtklib.h"
#include "rtklib_rtcm2.h"
#include "rtklib_rtcm3.h"
```

Macros

- #define RTCM2PREAMB 0x66
- #define RTCM3PREAMB 0xD3

Functions

- int **init_rtcm** (rtcm_t *rtcm)
- void **free_rtcm** (rtcm_t *rtcm)
- int **input_rtcm2** (rtcm_t *rtcm, unsigned char data)
- int **input_rtcm3** (rtcm_t *rtcm, unsigned char data)
- int **input_rtcm2f** (rtcm_t *rtcm, FILE *fp)
- int **input_rtcm3f** (rtcm_t *rtcm, FILE *fp)
- int **gen_rtcm2** (rtcm_t *rtcm, int type, int sync)

13.177.1 Detailed Description

RTCM functions headers.

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

SPDX-License-Identifier: BSD-2-Clause

Definition in file rtklib_rtcm.h.

13.177.2 Macro Definition Documentation

13.177.2.1 RTCM2PREAMB

```
#define RTCM2PREAMB 0x66
```

Definition at line 42 of file rtklib_rtcm.h.

13.177.2.2 RTCM3PREAMB

```
#define RTCM3PREAMB 0xD3
```

Definition at line 43 of file rtklib_rtcm.h.

13.178 rtklib_rtcm.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtcm.h
00003  * \brief RTCM functions headers
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * -----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * -----------------------------------------------------------------------------
00032  */
00033
00034
00035 #ifndef GNSS_SDR_RTKLIB_RTCM_H
00036 #define GNSS_SDR_RTKLIB_RTCM_H
00037
00038 #include "rtklib.h"
00039 #include "rtklib_rtcm2.h"
00040 #include "rtklib_rtcm3.h"
00041
```

```
00042 #define RTCM2PREAMB 0x66 /* rtcm ver.2 frame preamble */
00043 #define RTCM3PREAMB 0xD3 /* rtcm ver.3 frame preamble */
00044
00045
00046 int init_rtcm(rtcm_t *rtcm);
00047 void free_rtcm(rtcm_t *rtcm);
00048 int input_rtcm2(rtcm_t *rtcm, unsigned char data);
00049 int input_rtcm3(rtcm_t *rtcm, unsigned char data);
00050 int input_rtcm2f(rtcm_t *rtcm, FILE *fp);
00051 int input_rtcm3f(rtcm_t *rtcm, FILE *fp);
00052 int gen_rtcm2(rtcm_t *rtcm, int type, int sync);
00053 // int gen_rtcm3(rtcm_t *rtcm, int type, int sync);
00054
00055
00056 #endif  // GNSS_SDR_RTKLIB_RTCM_H
```

13.179 rtklib_rtcm2.h File Reference

RTCM v2 functions headers.
`#include "rtklib.h"`

Functions

- void **adjhour** (rtcm_t *rtcm, double zcnt)
- int **obsindex** (obs_t *obs, gtime_t time, int sat)
- int **decode_type1** (rtcm_t *rtcm)
- int **decode_type3** (rtcm_t *rtcm)
- int **decode_type14** (rtcm_t *rtcm, int ref_week=0)
- int **decode_type16** (rtcm_t *rtcm)
- int **decode_type17** (rtcm_t *rtcm, int ref_week=0)
- int **decode_type18** (rtcm_t *rtcm)
- int **decode_type19** (rtcm_t *rtcm)
- int **decode_type22** (rtcm_t *rtcm)
- int **decode_type23** (rtcm_t *rtcm)
- int **decode_type24** (rtcm_t *rtcm)
- int **decode_type31** (rtcm_t *rtcm)
- int **decode_type32** (rtcm_t *rtcm)
- int **decode_type34** (rtcm_t *rtcm)
- int **decode_type36** (rtcm_t *rtcm)
- int **decode_type37** (rtcm_t *rtcm)
- int **decode_type59** (rtcm_t *rtcm)
- int **decode_rtcm2** (rtcm_t *rtcm)

13.179.1 Detailed Description

RTCM v2 functions headers.

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_rtcm2.h.

13.180 rtklib_rtcm2.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtcm2.h
00003  * \brief RTCM v2 functions headers
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * ----------------------------------------------------------------------------
00032  */
00033
00034 #ifndef GNSS_SDR_RTKLIB_RTCM2_H
00035 #define GNSS_SDR_RTKLIB_RTCM2_H
00036
00037 #include "rtklib.h"
00038
00039
00040 void adjhour(rtcm_t *rtcm, double zcnt);
00041 int obsindex(obs_t *obs, gtime_t time, int sat);
00042 int decode_type1(rtcm_t *rtcm);
00043 int decode_type3(rtcm_t *rtcm);
00044 int decode_type14(rtcm_t *rtcm, int ref_week = 0);
00045 int decode_type16(rtcm_t *rtcm);
00046 int decode_type17(rtcm_t *rtcm, int ref_week = 0);
00047 int decode_type18(rtcm_t *rtcm);
00048 int decode_type19(rtcm_t *rtcm);
00049 int decode_type22(rtcm_t *rtcm);
00050 int decode_type23(rtcm_t *rtcm);
00051 int decode_type24(rtcm_t *rtcm);
00052 int decode_type31(rtcm_t *rtcm);
00053 int decode_type32(rtcm_t *rtcm);
00054 int decode_type34(rtcm_t *rtcm);
00055 int decode_type36(rtcm_t *rtcm);
00056 int decode_type37(rtcm_t *rtcm);
00057 int decode_type59(rtcm_t *rtcm);
00058 int decode_rtcm2(rtcm_t *rtcm);
00059
00060 #endif
```

13.181 rtklib_rtcm3.h File Reference

RTCM v3 functions headers.

`#include "rtklib.h"`

Functions

- double **getbitg** (const unsigned char *buff, int pos, int len)
- void **adjweek** (rtcm_t *rtcm, double tow)
- int **adjbdtweek** (int week)
- void **adjday_glot** (rtcm_t *rtcm, double tod)
- double **adjcp** (rtcm_t *rtcm, int sat, int freq, double cp)

- int **lossoflock** (rtcm_t *rtcm, int sat, int freq, int lock)
- unsigned char **snratio** (double snr)
- int **obsindex3** (obs_t *obs, gtime_t time, int sat)
- int **test_staid** (rtcm_t *rtcm, int staid)
- int **decode_head1001** (rtcm_t *rtcm, int *sync)
- int **decode_type1001** (rtcm_t *rtcm)
- int **decode_type1002** (rtcm_t *rtcm)
- int **decode_type1003** (rtcm_t *rtcm)
- int **decode_type1004** (rtcm_t *rtcm)
- double **getbits_38** (const unsigned char *buff, int pos)
- int **decode_type1005** (rtcm_t *rtcm)
- int **decode_type1006** (rtcm_t *rtcm)
- int **decode_type1007** (rtcm_t *rtcm)
- int **decode_type1008** (rtcm_t *rtcm)
- int **decode_head1009** (rtcm_t *rtcm, int *sync)
- int **decode_type1009** (rtcm_t *rtcm)
- int **decode_type1010** (rtcm_t *rtcm)
- int **decode_type1011** (rtcm_t *rtcm)
- int **decode_type1012** (rtcm_t *rtcm)
- int **decode_type1013** (rtcm_t *rtcm)
- int **decode_type1019** (rtcm_t *rtcm, int ref_week=0)
- int **decode_type1020** (rtcm_t *rtcm)
- int **decode_type1021** (rtcm_t *rtcm)
- int **decode_type1022** (rtcm_t *rtcm)
- int **decode_type1023** (rtcm_t *rtcm)
- int **decode_type1024** (rtcm_t *rtcm)
- int **decode_type1025** (rtcm_t *rtcm)
- int **decode_type1026** (rtcm_t *rtcm)
- int **decode_type1027** (rtcm_t *rtcm)
- int **decode_type1029** (rtcm_t *rtcm)
- int **decode_type1030** (rtcm_t *rtcm)
- int **decode_type1031** (rtcm_t *rtcm)
- int **decode_type1032** (rtcm_t *rtcm)
- int **decode_type1033** (rtcm_t *rtcm)
- int **decode_type1034** (rtcm_t *rtcm)
- int **decode_type1035** (rtcm_t *rtcm)
- int **decode_type1037** (rtcm_t *rtcm)
- int **decode_type1038** (rtcm_t *rtcm)
- int **decode_type1039** (rtcm_t *rtcm)
- int **decode_type1044** (rtcm_t *rtcm, int ref_week=0)
- int **decode_type1045** (rtcm_t *rtcm)
- int **decode_type1046** (rtcm_t *rtcm)
- int **decode_type1047** (rtcm_t *rtcm)
- int **decode_type1063** (rtcm_t *rtcm)
- int **decode_ssr1_head** (rtcm_t *rtcm, int sys, int *sync, int *iod, double *udint, int *refd, int *hsize)
- int **decode_ssr2_head** (rtcm_t *rtcm, int sys, int *sync, int *iod, double *udint, int *hsize)
- int **decode_ssr7_head** (rtcm_t *rtcm, int sys, int *sync, int *iod, double *udint, int *dispe, int *mw, int *hsize)
- int **decode_ssr1** (rtcm_t *rtcm, int sys)
- int **decode_ssr2** (rtcm_t *rtcm, int sys)
- int **decode_ssr3** (rtcm_t *rtcm, int sys)
- int **decode_ssr4** (rtcm_t *rtcm, int sys)
- int **decode_ssr5** (rtcm_t *rtcm, int sys)
- int **decode_ssr6** (rtcm_t *rtcm, int sys)
- int **decode_ssr7** (rtcm_t *rtcm, int sys)
- void **sigindex** (int sys, const unsigned char *code, const int *freq, int n, const char *opt, int *ind)

- void **save_msm_obs** (rtcm_t *rtcm, int sys, msm_h_t *h, const double *r, const double *pr, const double *cp, const double *rr, const double *rrf, const double *cnr, const int *lock, const int *ex, const int *half)
- int **decode_msm_head** (rtcm_t *rtcm, int sys, int *sync, int *iod, msm_h_t *h, int *hsize)
- int **decode_msm0** (rtcm_t *rtcm, int sys)
- int **decode_msm4** (rtcm_t *rtcm, int sys)
- int **decode_msm5** (rtcm_t *rtcm, int sys)
- int **decode_msm6** (rtcm_t *rtcm, int sys)
- int **decode_msm7** (rtcm_t *rtcm, int sys)
- int **decode_type1230** (rtcm_t *rtcm)
- int **decode_rtcm3** (rtcm_t *rtcm)

Variables

- const double PRUNIT_GPS = 299792.458
- const double PRUNIT_GLO = 599584.916
- const double RANGE_MS = SPEED_OF_LIGHT_M_S * 0.001
- const double SSRUDINT [16]
- const int CODES_GPS [ ]
- const int CODES_GLO [ ]
- const int CODES_GAL [ ]
- const int CODES_QZS [ ]
- const int CODES_BDS [ ]
- const int CODES_SBS [ ]

13.181.1 Detailed Description

RTCM v3 functions headers.

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

SPDX-License-Identifier: BSD-2-Clause

Definition in file rtklib_rtcm3.h.

13.181.2 Variable Documentation

13.181.2.1 CODES_BDS

```
const int CODES_BDS[]
```

Initial value:

```
= {
CODE_L2I, CODE_L2Q, CODE_L2X, CODE_L7I, CODE_L7Q, CODE_L7X, CODE_L6I, CODE_L6Q,
CODE_L6X}
```

Definition at line 73 of file rtklib_rtcm3.h.

13.181.2.2 CODES_GAL

```
const int CODES_GAL[]
```

Initial value:

```
                = {
CODE_L1A, CODE_L1B, CODE_L1C, CODE_L1X, CODE_L1Z, CODE_L5I, CODE_L5Q, CODE_L5X,
CODE_L7I, CODE_L7Q, CODE_L7X, CODE_L8I, CODE_L8Q, CODE_L8X, CODE_L6A, CODE_L6B,
CODE_L6C, CODE_L6X, CODE_L6Z}
```

Definition at line 62 of file rtklib_rtcm3.h.

13.181.2.3 CODES_GLO

```
const int CODES_GLO[]
```

Initial value:

```
                = {
CODE_L1C, CODE_L1P, CODE_L2C, CODE_L2P}
```

Definition at line 58 of file rtklib_rtcm3.h.

13.181.2.4 CODES_GPS

```
const int CODES_GPS[]
```

Initial value:

```
                = {
CODE_L1C, CODE_L1P, CODE_L1W, CODE_L1Y, CODE_L1M, CODE_L2C, CODE_L2D, CODE_L2S,
CODE_L2L, CODE_L2X, CODE_L2P, CODE_L2W, CODE_L2Y, CODE_L2M, CODE_L5I, CODE_L5Q,
CODE_L5X}
```

Definition at line 52 of file rtklib_rtcm3.h.

13.181.2.5 CODES_QZS

```
const int CODES_QZS[]
```

Initial value:

```
                = {
CODE_L1C, CODE_L1S, CODE_L1L, CODE_L2S, CODE_L2L, CODE_L2X, CODE_L5I, CODE_L5Q,
CODE_L5X, CODE_L6S, CODE_L6L, CODE_L6X, CODE_L1X}
```

Definition at line 68 of file rtklib_rtcm3.h.

13.181.2.6 CODES_SBS

```
const int CODES_SBS[]
```

Initial value:

```
                = {
CODE_L1C, CODE_L5I, CODE_L5Q, CODE_L5X}
```

Definition at line 78 of file rtklib_rtcm3.h.

13.181.2.7 PRUNIT_GLO

```
const double PRUNIT_GLO = 599584.916
```

Definition at line 42 of file rtklib_rtcm3.h.

13.181.2.8 PRUNIT_GPS

```
const double PRUNIT_GPS = 299792.458
```

Definition at line 41 of file rtklib_rtcm3.h.

13.181.2.9 RANGE_MS

```
const double RANGE_MS = SPEED_OF_LIGHT_M_S * 0.001
```

Definition at line 43 of file rtklib_rtcm3.h.

13.181.2.10 SSRUDINT

```
const double SSRUDINT[16]
```

Initial value:

```
                = {
1, 2, 5, 10, 15, 30, 60, 120, 240, 300, 600, 900, 1800, 3600, 7200, 10800}
```

Definition at line 47 of file rtklib_rtcm3.h.

13.182 rtklib_rtcm3.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtcm3.h
00003  * \brief RTCM v3 functions headers
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * ----------------------------------------------------------------------------
00032  */
00033
00034 #ifndef GNSS_SDR_RTKLIB_RTCM3_H
00035 #define GNSS_SDR_RTKLIB_RTCM3_H
00036
00037 #include "rtklib.h"
00038
00039 /* constants -----------------------------------------------------------------*/
00040
00041 const double PRUNIT_GPS = 299792.458;               /* rtcm ver.3 unit of gps pseudorange (m) */
00042 const double PRUNIT_GLO = 599584.916;               /* rtcm ver.3 unit of glonass pseudorange (m) */
00043 const double RANGE_MS = SPEED_OF_LIGHT_M_S * 0.001; /* range in 1 ms */
00044
00045
00046 /* ssr update intervals ------------------------------------------------------*/
00047 const double SSRUDINT[16] = {
00048     1, 2, 5, 10, 15, 30, 60, 120, 240, 300, 600, 900, 1800, 3600, 7200, 10800};
00049
00050
00051 /* ssr 3 and 7 signal and tracking mode ids ----------------------------------*/
00052 const int CODES_GPS[] = {
00053     CODE_L1C, CODE_L1P, CODE_L1W, CODE_L1Y, CODE_L1M, CODE_L2C, CODE_L2D, CODE_L2S,
00054     CODE_L2L, CODE_L2X, CODE_L2P, CODE_L2W, CODE_L2Y, CODE_L2M, CODE_L5I, CODE_L5Q,
00055     CODE_L5X};
00056
00057
00058 const int CODES_GLO[] = {
00059     CODE_L1C, CODE_L1P, CODE_L2C, CODE_L2P};
00060
00061
00062 const int CODES_GAL[] = {
00063     CODE_L1A, CODE_L1B, CODE_L1C, CODE_L1X, CODE_L1Z, CODE_L5I, CODE_L5Q, CODE_L5X,
00064     CODE_L7I, CODE_L7Q, CODE_L7X, CODE_L8I, CODE_L8Q, CODE_L8X, CODE_L6A, CODE_L6B,
00065     CODE_L6C, CODE_L6X, CODE_L6Z};
00066
00067
00068 const int CODES_QZS[] = {
00069     CODE_L1C, CODE_L1S, CODE_L1L, CODE_L2S, CODE_L2L, CODE_L2X, CODE_L5I, CODE_L5Q,
00070     CODE_L5X, CODE_L6S, CODE_L6L, CODE_L6X, CODE_L1X};
00071
```

```
00072
00073 const int CODES_BDS[] = {
00074     CODE_L2I, CODE_L2Q, CODE_L2X, CODE_L7I, CODE_L7Q, CODE_L7X, CODE_L6I, CODE_L6Q,
00075     CODE_L6X};
00076
00077
00078 const int CODES_SBS[] = {
00079     CODE_L1C, CODE_L5I, CODE_L5Q, CODE_L5X};
00080
00081
00082 double getbitg(const unsigned char *buff, int pos, int len);
00083
00084 void adjweek(rtcm_t *rtcm, double tow);
00085
00086 int adjbdtweek(int week);
00087
00088 void adjday_glot(rtcm_t *rtcm, double tod);
00089
00090 double adjcp(rtcm_t *rtcm, int sat, int freq, double cp);
00091
00092 int lossoflock(rtcm_t *rtcm, int sat, int freq, int lock);
00093
00094 unsigned char snratio(double snr);
00095
00096 int obsindex3(obs_t *obs, gtime_t time, int sat);
00097
00098 int test_staid(rtcm_t *rtcm, int staid);
00099
00100 int decode_head1001(rtcm_t *rtcm, int *sync);
00101
00102 int decode_type1001(rtcm_t *rtcm);
00103
00104 int decode_type1002(rtcm_t *rtcm);
00105
00106 int decode_type1003(rtcm_t *rtcm);
00107
00108 int decode_type1004(rtcm_t *rtcm);
00109
00110 double getbits_38(const unsigned char *buff, int pos);
00111
00112 int decode_type1005(rtcm_t *rtcm);
00113
00114 int decode_type1006(rtcm_t *rtcm);
00115
00116 int decode_type1007(rtcm_t *rtcm);
00117
00118 int decode_type1008(rtcm_t *rtcm);
00119
00120 int decode_head1009(rtcm_t *rtcm, int *sync);
00121
00122 int decode_type1009(rtcm_t *rtcm);
00123
00124 int decode_type1010(rtcm_t *rtcm);
00125
00126 int decode_type1011(rtcm_t *rtcm);
00127
00128 int decode_type1012(rtcm_t *rtcm);
00129
00130 int decode_type1013(rtcm_t *rtcm);
00131
00132 int decode_type1019(rtcm_t *rtcm, int ref_week = 0);
00133
00134 int decode_type1020(rtcm_t *rtcm);
00135
00136 int decode_type1021(rtcm_t *rtcm);
00137
00138 int decode_type1022(rtcm_t *rtcm);
00139
00140 int decode_type1023(rtcm_t *rtcm);
00141
00142 int decode_type1024(rtcm_t *rtcm);
00143
00144 int decode_type1025(rtcm_t *rtcm);
00145
00146 int decode_type1026(rtcm_t *rtcm);
00147
00148 int decode_type1027(rtcm_t *rtcm);
00149
00150 int decode_type1029(rtcm_t *rtcm);
00151
00152 int decode_type1030(rtcm_t *rtcm);
00153
00154 int decode_type1031(rtcm_t *rtcm);
00155
00156 int decode_type1032(rtcm_t *rtcm);
00157
00158 int decode_type1033(rtcm_t *rtcm);
```

```
00159
00160 int decode_type1034(rtcm_t *rtcm);
00161
00162 int decode_type1035(rtcm_t *rtcm);
00163
00164 int decode_type1037(rtcm_t *rtcm);
00165
00166 int decode_type1038(rtcm_t *rtcm);
00167
00168 int decode_type1039(rtcm_t *rtcm);
00169
00170 int decode_type1044(rtcm_t *rtcm, int ref_week = 0);
00171
00172 int decode_type1045(rtcm_t *rtcm);
00173
00174 int decode_type1046(rtcm_t *rtcm);
00175
00176 int decode_type1047(rtcm_t *rtcm);
00177
00178 int decode_type1063(rtcm_t *rtcm);
00179
00180 int decode_ssr1_head(rtcm_t *rtcm, int sys, int *sync, int *iod,
00181     double *udint, int *refd, int *hsize);
00182
00183 int decode_ssr2_head(rtcm_t *rtcm, int sys, int *sync, int *iod,
00184     double *udint, int *hsize);
00185
00186 int decode_ssr7_head(rtcm_t *rtcm, int sys, int *sync, int *iod,
00187     double *udint, int *dispe, int *mw, int *hsize);
00188
00189 int decode_ssr1(rtcm_t *rtcm, int sys);
00190
00191 int decode_ssr2(rtcm_t *rtcm, int sys);
00192
00193 int decode_ssr3(rtcm_t *rtcm, int sys);
00194
00195 int decode_ssr4(rtcm_t *rtcm, int sys);
00196
00197 int decode_ssr5(rtcm_t *rtcm, int sys);
00198
00199 int decode_ssr6(rtcm_t *rtcm, int sys);
00200
00201 int decode_ssr7(rtcm_t *rtcm, int sys);
00202
00203 void sigindex(int sys, const unsigned char *code, const int *freq, int n,
00204     const char *opt, int *ind);
00205
00206 void save_msm_obs(rtcm_t *rtcm, int sys, msm_h_t *h, const double *r,
00207     const double *pr, const double *cp, const double *rr,
00208     const double *rrf, const double *cnr, const int *lock,
00209     const int *ex, const int *half);
00210
00211 int decode_msm_head(rtcm_t *rtcm, int sys, int *sync, int *iod,
00212     msm_h_t *h, int *hsize);
00213
00214 int decode_msm0(rtcm_t *rtcm, int sys);
00215
00216 int decode_msm4(rtcm_t *rtcm, int sys);
00217
00218 int decode_msm5(rtcm_t *rtcm, int sys);
00219
00220 int decode_msm6(rtcm_t *rtcm, int sys);
00221
00222 int decode_msm7(rtcm_t *rtcm, int sys);
00223
00224 int decode_type1230(rtcm_t *rtcm);
00225
00226 int decode_rtcm3(rtcm_t *rtcm);
00227
00228
00229 #endif
```

13.183 rtklib_rtkcmn.h File Reference

rtklib common functions

```
#include "rtklib.h"
#include <cstddef>
#include <string>
#include <boost/filesystem/operations.hpp>
#include <boost/filesystem/path.hpp>
#include <boost/filesystem/path_traits.hpp>
```

```
#include <boost/system/error_code.hpp>
```

Macros

- #define Rx(t, X)
- #define Ry(t, X)
- #define Rz(t, X)

Functions

- char * **strncpy_no_trunc** (char *out, size_t outsz, const char *in, size_t insz)
- void **fatalerr** (const char *format,...)
- int **satno** (int sys, int prn)
- int **satsys** (int sat, int *prn)
- int **satid2no** (const char *id)
- std::string **satno2id** (int sat)
- int **satexclude** (int sat, int svh, const prcopt_t *opt)
- int **testsnr** (int base, int freq, double el, double snr, const snrmask_t *mask)
- unsigned char **obs2code** (const char *obs, int *freq)
- char * **code2obs** (unsigned char code, int *freq)
- bool **is_bds_b1c_code** (unsigned char code)
- bool **is_bds_b2a_code** (unsigned char code)
- void **bds_observation_slots** (const obsd_t *obs, int *primary, int *secondary)
- void **setcodepri** (int sys, int freq, const char *pri)
- int **getcodepri** (int sys, unsigned char code, const char *opt)
- unsigned int **getbitu** (const unsigned char *buff, int pos, int len)
- int **getbits** (const unsigned char *buff, int pos, int len)
- void **setbitu** (unsigned char *buff, int pos, int len, unsigned int data)
- void **setbits** (unsigned char *buff, int pos, int len, int data)
- unsigned int **rtk_crc32** (const unsigned char *buff, int len)
- unsigned int **rtk_crc24q** (const unsigned char *buff, int len)
- unsigned short **rtk_crc16** (const unsigned char *buff, int len)
- int **decode_word** (unsigned int word, unsigned char *data)
- double * **mat** (int n, int m)
- int * **imat** (int n, int m)
- double * **zeros** (int n, int m)
- double * **eye** (int n)
- double **dot** (const double *a, const double *b, int n)
- double **norm_rtk** (const double *a, int n)
- void **cross3** (const double *a, const double *b, double *c)
- int **normv3** (const double *a, double *b)
- void **matcpy** (double *A, const double *B, int n, int m)
- void **matmul** (const char *tr, int n, int k, int m, double alpha, const double *A, const double *B, double beta, double *C)
- int **matinv** (double *A, int n)
- int **solve** (const char *tr, const double *A, const double *Y, int n, int m, double *X)
- int **lsq** (const double *A, const double *y, int n, int m, double *x, double *Q)
- int **filter_** (const double *x, const double *P, const double *H, const double *v, const double *R, int n, int m, double *xp, double *Pp)
- int **filter** (double *x, double *P, const double *H, const double *v, const double *R, int n, int m)
- int **smoother** (const double *xf, const double *Qf, const double *xb, const double *Qb, int n, double *xs, double *Qs)
- void **matfprint** (const double A[ ], int n, int m, int p, int q, FILE *fp)
- void **matsprint** (const double A[ ], int n, int m, int p, int q, std::string &buffer)
- void **matprint** (const double A[ ], int n, int m, int p, int q)

- double **str2num** (const char *s, int i, int n)
- int **str2time** (const char *s, int i, int n, gtime_t *t)
- gtime_t **epoch2time** (const double *ep)
- void **time2epoch** (gtime_t t, double *ep)
- gtime_t **gpst2time** (int week, double sec)
- double **time2gpst** (gtime_t t, int *week)
- gtime_t **gst2time** (int week, double sec)
- double **time2gst** (gtime_t t, int *week)
- gtime_t **bdt2time** (int week, double sec)
- double **time2bdt** (gtime_t t, int *week)
- gtime_t **timeadd** (gtime_t t, double sec)
- double **timediff** (gtime_t t1, gtime_t t2)
- double **timediffweekcrossover** (gtime_t t1, gtime_t t2)
- gtime_t **timeget** ()
- void **timeset** (gtime_t t)
- int **read_leaps_text** (FILE *fp)
- int **read_leaps_usno** (FILE *fp)
- int **read_leaps** (const char *file)
- gtime_t **gpst2utc** (gtime_t t)
- gtime_t **utc2gpst** (gtime_t t)
- gtime_t **gpst2bdt** (gtime_t t)
- gtime_t **bdt2gpst** (gtime_t t)
- double **time2sec** (gtime_t time, gtime_t *day)
- double **utc2gmst** (gtime_t t, double ut1_utc)
- void **time2str** (gtime_t t, char *s, int n)
- char * **time_str** (gtime_t t, int n)
- double **time2doy** (gtime_t t)
- int **adjgpsweek** (int week, int ref_week=0)
- unsigned int **tickget** ()
- void **sleepms** (int ms)
- void **deg2dms** (double deg, double *dms, int ndec)
- void **deg2dms** (double deg, double *dms)
- double **dms2deg** (const double *dms)
- void **ecef2pos** (const double *r, double *pos)
- void **pos2ecef** (const double *pos, double *r)
- void **xyz2enu** (const double *pos, double *E)
- void **ecef2enu** (const double *pos, const double *r, double *e)
- void **enu2ecef** (const double *pos, const double *e, double *r)
- void **sta2antpos** (const sta_t *sta, double *rr)
- void **covenu** (const double *pos, const double *P, double *Q)
- void **covecef** (const double *pos, const double *Q, double *P)
- void **ast_args** (double t, double *f)
- void **nut_iau1980** (double t, const double *f, double *dpsi, double *deps)
- void **eci2ecef** (gtime_t tutc, const double *erpv, double *U, double *gmst)
- int **decodef** (char *p, int n, double *v)
- void **addpcv** (const pcv_t *pcv, pcvs_t *pcvs)
- int **readngspcv** (const char *file, pcvs_t *pcvs)
- int **readantex** (const char *file, pcvs_t *pcvs)
- int **readpcv** (const char *file, pcvs_t *pcvs)
- pcv_t * **searchpcv** (int sat, const char *type, gtime_t time, const pcvs_t *pcvs)
- void **readpos** (const char *file, const char *rcv, double *pos)
- int **readblqrecord** (FILE *fp, double *odisp)
- int **readblq** (const char *file, const char *sta, double *odisp)
- int **readerp** (const char *file, erp_t *erp)
- int **geterp** (const erp_t *erp, gtime_t time, double *erpv)

- int **cmpeph** (const void *p1, const void *p2)
- void **uniqeph** (nav_t *nav)
- int **cmpgeph** (const void *p1, const void *p2)
- void **uniqgeph** (nav_t *nav)
- int **cmpseph** (const void *p1, const void *p2)
- void **uniqseph** (nav_t *nav)
- void **uniqnav** (nav_t *nav)
- int **cmpobs** (const void *p1, const void *p2)
- int **sortobs** (obs_t *obs)
- int **screent** (gtime_t time, gtime_t ts, gtime_t te, double tint)
- int **readnav** (const char *file, nav_t *nav)
- int **savenav** (const char *file, const nav_t *nav)
- void **freeobs** (obs_t *obs)
- void **freenav** (nav_t *nav, int opt)
- void **traceopen** (const char *file)
- void **traceclose** ()
- void **tracelevel** (int level)
- void **traceswap** ()
- void **trace** (int level, const char *format,...)
- void **tracet** (int level, const char *format,...)
- void **tracemat** (int level, const double *A, int n, int m, int p, int q)
- void **traceobs** (int level, const obsd_t *obs, int n)
- int **execcmd** (const char *cmd)
- void **createdir** (fs::path const &path)
- int **reppath** (std::string const &path, std::string &rpath, gtime_t time, const char *rov, const char *base)
- double **satwavelen** (int sat, int frq, const nav_t *nav)
- double **geodist** (const double *rs, const double *rr, double *e)
- double **satazel** (const double *pos, const double *e, double *azel)
- void **dops** (int ns, const double *azel, double elmin, double *dop)
- double **ionmodel** (gtime_t t, const double *ion, const double *pos, const double *azel)
- double **ionmapf** (const double *pos, const double *azel)
- double **ionppp** (const double *pos, const double *azel, double re, double hion, double *posp)
- double **tropmodel** (gtime_t time, const double *pos, const double *azel, double humi)
- double **interpc** (const double coef[ ], double lat)
- double **mapf** (double el, double a, double b, double c)
- double **nmf** (gtime_t time, const double pos[ ], const double azel[ ], double *mapfw)
- double **tropmapf** (gtime_t time, const double pos[ ], const double azel[ ], double *mapfw)
- double **interpvar** (double ang, const double *var)
- void **antmodel** (const pcv_t *pcv, const double *del, const double *azel, int opt, double *dant)
- void **antmodel_s** (const pcv_t *pcv, double nadir, double *dant)
- void **sunmoonpos_eci** (gtime_t tut, double *rsun, double *rmoon)
- void **sunmoonpos** (gtime_t tutc, const double *erpv, double *rsun, double *rmoon, double *gmst)
- void **csmooth** (obs_t *obs, int ns)
- int **rtk_uncompress** (const char *file, char *uncfile)
- int **expath** (const char *path, char *paths[ ], int nmax)
- void **windupcorr** (gtime_t time, const double *rs, const double *rr, double *phw)

13.183.1 Detailed Description

rtklib common functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References : [1] IS-GPS-200M, Navstar GPS Space Segment/Navigation User Interfaces, May, 2021 [2] RTCA/DO-229C, Minimum operational performance standards for global positioning system/wide area augmentation system airborne equipment, RTCA inc, November 28, 2001 [3] M.Rothacher, R.Schmid, ANTEX: The Antenna Exchange Format Version 1.4, 15 September, 2010 [4] A.Gelb ed., Applied Optimal Estimation, The M.I.T Press, 1974 [5] A.E.Niell, Global mapping functions for the atmosphere delay at radio wavelengths, Journal of geophysical research, 1996 [6] W.Gurtner and L.Estey, RINEX The Receiver Independent Exchange Format Version 3.00, November 28, 2007 [7] J.Kouba, A Guide to using International GNSS Service (IGS) products, May 2009 [8] China Satellite Navigation Office, BeiDou navigation satellite system signal in space interface control document, open service signal B1I (version 1.0), Dec 2012 [9] J.Boehm, A.Niell, P.Tregoning and H.Shuh, Global Mapping Function (GMF): A new empirical mapping function base on numerical weather model data, Geophysical Research Letters, 33, L07304, 2006 [10] GLONASS/GPS/Galileo/Compass/SBAS NV08C receiver series BINR interface protocol specification ver.1.3, August, 2012
Definition in file rtklib_rtkcmn.h.

13.183.2 Macro Definition Documentation

13.183.2.1 Rx

```
#define Rx(
                t,
                X)
```

Value:

```
do                                                  \
    {                                               \
        (X)[0] = 1.0;                               \
        (X)[1] = (X)[2] = (X)[3] = (X)[6] = 0.0;    \
        (X)[4] = (X)[8] = cos(t);                   \
        (X)[7] = sin(t);                            \
        (X)[5] = -(X)[7];                           \
    }                                               \
while (0)
```

Definition at line 85 of file rtklib_rtkcmn.h.

13.183.2.2 Ry

```
#define Ry(
                t,
                X)
```

Value:

```
do                                                  \
    {                                               \
        (X)[4] = 1.0;                               \
        (X)[1] = (X)[3] = (X)[5] = (X)[7] = 0.0;    \
        (X)[0] = (X)[8] = cos(t);                   \
        (X)[2] = sin(t);                            \
        (X)[6] = -(X)[2];                           \
    }                                               \
```

```
while (0)
```

Definition at line 96 of file rtklib_rtkcmn.h.

13.183.2.3 Rz

```
#define Rz(
              t,
              X)
```

Value:

```
do                                                  \
    {                                               \
        (X)[8] = 1.0;                               \
        (X)[2] = (X)[5] = (X)[6] = (X)[7] = 0.0;    \
        (X)[0] = (X)[4] = cos(t);                   \
        (X)[3] = sin(t);                            \
        (X)[1] = -(X)[3];                           \
    }                                               \
while (0)
```

Definition at line 107 of file rtklib_rtkcmn.h.

13.184 rtklib_rtkcmn.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtkcmn.h
00003  * \brief rtklib common functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * References :
00032  *     [1] IS-GPS-200M, Navstar GPS Space Segment/Navigation User Interfaces,
00033  *         May, 2021
00034  *     [2] RTCA/DO-229C, Minimum operational performance standards for global
00035  *         positioning system/wide area augmentation system airborne equipment,
00036  *         RTCA inc, November 28, 2001
00037  *     [3] M.Rothacher, R.Schmid, ANTEX: The Antenna Exchange Format Version 1.4,
00038  *         15 September, 2010
00039  *     [4] A.Gelb ed., Applied Optimal Estimation, The M.I.T Press, 1974
00040  *     [5] A.E.Niell, Global mapping functions for the atmosphere delay at radio
00041  *         wavelengths, Journal of geophysical research, 1996
00042  *     [6] W.Gurtner and L.Estey, RINEX The Receiver Independent Exchange Format
00043  *         Version 3.00, November 28, 2007
00044  *     [7] J.Kouba, A Guide to using International GNSS Service (IGS) products,
00045  *         May 2009
00046  *     [8] China Satellite Navigation Office, BeiDou navigation satellite system
00047  *         signal in space interface control document, open service signal B1I
00048  *         (version 1.0), Dec 2012
00049  *     [9] J.Boehm, A.Niell, P.Tregoning and H.Shuh, Global Mapping Function
00050  *         (GMF): A new empirical mapping function base on numerical weather
00051  *         model data, Geophysical Research Letters, 33, L07304, 2006
00052  *     [10] GLONASS/GPS/Galileo/Compass/SBAS NV08C receiver series BINR interface
00053  *         protocol specification ver.1.3, August, 2012
00054  *
00055  * ----------------------------------------------------------------------------
00056  */
```

```
00057
00058 #ifndef GNSS_SDR_RTKLIB_RTKCMN_H
00059 #define GNSS_SDR_RTKLIB_RTKCMN_H
00060
00061 #include "rtklib.h"
00062 #include <cstddef>
00063 #include <string>
00064
00065 #if HAS_STD_FILESYSTEM
00066 #include <system_error>
00067 namespace errorlib = std;
00068 #if HAS_STD_FILESYSTEM_EXPERIMENTAL
00069 #include <experimental/filesystem>
00070 namespace fs = std::experimental::filesystem;
00071 #else
00072 #include <filesystem>
00073 namespace fs = std::filesystem;
00074 #endif
00075 #else
00076 #include <boost/filesystem/operations.hpp>   // for create_directories, exists
00077 #include <boost/filesystem/path.hpp>         // for path, operator«
00078 #include <boost/filesystem/path_traits.hpp>  // for filesystem
00079 #include <boost/system/error_code.hpp>       // for error_code
00080 namespace fs = boost::filesystem;
00081 namespace errorlib = boost::system;
00082 #endif
00083
00084 /* coordinate rotation matrix ---------------------------------------------*/
00085 #define Rx(t, X)                                        \
00086     do                                                  \
00087         {                                               \
00088             (X)[0] = 1.0;                               \
00089             (X)[1] = (X)[2] = (X)[3] = (X)[6] = 0.0; \
00090             (X)[4] = (X)[8] = cos(t);                   \
00091             (X)[7] = sin(t);                            \
00092             (X)[5] = -(X)[7];                           \
00093         }                                               \
00094     while (0)
00095
00096 #define Ry(t, X)                                        \
00097     do                                                  \
00098         {                                               \
00099             (X)[4] = 1.0;                               \
00100             (X)[1] = (X)[3] = (X)[5] = (X)[7] = 0.0; \
00101             (X)[0] = (X)[8] = cos(t);                   \
00102             (X)[2] = sin(t);                            \
00103             (X)[6] = -(X)[2];                           \
00104         }                                               \
00105     while (0)
00106
00107 #define Rz(t, X)                                        \
00108     do                                                  \
00109         {                                               \
00110             (X)[8] = 1.0;                               \
00111             (X)[2] = (X)[5] = (X)[6] = (X)[7] = 0.0; \
00112             (X)[0] = (X)[4] = cos(t);                   \
00113             (X)[3] = sin(t);                            \
00114             (X)[1] = -(X)[3];                           \
00115         }                                               \
00116     while (0)
00117
00118 char *strncpy_no_trunc(char *out, size_t outsz, const char *in, size_t insz);
00119 void fatalerr(const char *format, ...);
00120 int satno(int sys, int prn);
00121 int satsys(int sat, int *prn);
00122 int satid2no(const char *id);
00123 std::string satno2id(int sat);
00124 int satexclude(int sat, int svh, const prcopt_t *opt);
00125 int testsnr(int base, int freq, double el, double snr, const snrmask_t *mask);
00126 unsigned char obs2code(const char *obs, int *freq);
00127 char *code2obs(unsigned char code, int *freq);
00128 bool is_bds_b1c_code(unsigned char code);
00129 bool is_bds_b2a_code(unsigned char code);
00130 void bds_observation_slots(const obsd_t *obs, int *primary, int *secondary);
00131 void setcodepri(int sys, int freq, const char *pri);
00132 int getcodepri(int sys, unsigned char code, const char *opt);
00133 unsigned int getbitu(const unsigned char *buff, int pos, int len);
00134 int getbits(const unsigned char *buff, int pos, int len);
00135 void setbitu(unsigned char *buff, int pos, int len, unsigned int data);
00136 void setbits(unsigned char *buff, int pos, int len, int data);
00137 unsigned int rtk_crc32(const unsigned char *buff, int len);
00138 unsigned int rtk_crc24q(const unsigned char *buff, int len);
00139 unsigned short rtk_crc16(const unsigned char *buff, int len);
00140 int decode_word(unsigned int word, unsigned char *data);
00141 double *mat(int n, int m);
00142 int *imat(int n, int m);
00143 double *zeros(int n, int m);
```

```
00144 double *eye(int n);
00145 double dot(const double *a, const double *b, int n);
00146 double norm_rtk(const double *a, int n);
00147 void cross3(const double *a, const double *b, double *c);
00148 int normv3(const double *a, double *b);
00149 void matcpy(double *A, const double *B, int n, int m);
00150 void matmul(const char *tr, int n, int k, int m, double alpha,
00151     const double *A, const double *B, double beta, double *C);
00152 int matinv(double *A, int n);
00153 int solve(const char *tr, const double *A, const double *Y, int n,
00154     int m, double *X);
00155 int lsq(const double *A, const double *y, int n, int m, double *x,
00156     double *Q);
00157 int filter_(const double *x, const double *P, const double *H,
00158     const double *v, const double *R, int n, int m,
00159     double *xp, double *Pp);
00160 int filter(double *x, double *P, const double *H, const double *v,
00161     const double *R, int n, int m);
00162 int smoother(const double *xf, const double *Qf, const double *xb,
00163     const double *Qb, int n, double *xs, double *Qs);
00164 void matfprint(const double A[], int n, int m, int p, int q, FILE *fp);
00165 void matsprint(const double A[], int n, int m, int p, int q, std::string &buffer);
00166 void matprint(const double A[], int n, int m, int p, int q);
00167 double str2num(const char *s, int i, int n);
00168 int str2time(const char *s, int i, int n, gtime_t *t);
00169 gtime_t epoch2time(const double *ep);
00170 void time2epoch(gtime_t t, double *ep);
00171 gtime_t gpst2time(int week, double sec);
00172 double time2gpst(gtime_t t, int *week);
00173 gtime_t gst2time(int week, double sec);
00174 double time2gst(gtime_t t, int *week);
00175 gtime_t bdt2time(int week, double sec);
00176 double time2bdt(gtime_t t, int *week);
00177 gtime_t timeadd(gtime_t t, double sec);
00178 double timediff(gtime_t t1, gtime_t t2);
00179 double timediffweekcrossover(gtime_t t1, gtime_t t2);
00180 gtime_t timeget();
00181 void timeset(gtime_t t);
00182 int read_leaps_text(FILE *fp);
00183 int read_leaps_usno(FILE *fp);
00184 int read_leaps(const char *file);
00185 gtime_t gpst2utc(gtime_t t);
00186 gtime_t utc2gpst(gtime_t t);
00187 gtime_t gpst2bdt(gtime_t t);
00188 gtime_t bdt2gpst(gtime_t t);
00189 double time2sec(gtime_t time, gtime_t *day);
00190 double utc2gmst(gtime_t t, double ut1_utc);
00191 void time2str(gtime_t t, char *s, int n);
00192 char *time_str(gtime_t t, int n);
00193 double time2doy(gtime_t t);
00194 int adjgpsweek(int week, int ref_week = 0);
00195 unsigned int tickget();
00196 void sleepms(int ms);
00197 void deg2dms(double deg, double *dms, int ndec);
00198 void deg2dms(double deg, double *dms);
00199 double dms2deg(const double *dms);
00200 void ecef2pos(const double *r, double *pos);
00201 void pos2ecef(const double *pos, double *r);
00202 void xyz2enu(const double *pos, double *E);
00203 void ecef2enu(const double *pos, const double *r, double *e);
00204 void enu2ecef(const double *pos, const double *e, double *r);
00205
00206 void sta2antpos(const sta_t *sta, double *rr);
00207 void covenu(const double *pos, const double *P, double *Q);
00208 void covecef(const double *pos, const double *Q, double *P);
00209 void ast_args(double t, double *f);
00210 void nut_iau1980(double t, const double *f, double *dpsi, double *deps);
00211 void eci2ecef(gtime_t tutc, const double *erpv, double *U, double *gmst);
00212 int decodef(char *p, int n, double *v);
00213 void addpcv(const pcv_t *pcv, pcvs_t *pcvs);
00214 int readngspcv(const char *file, pcvs_t *pcvs);
00215 int readantex(const char *file, pcvs_t *pcvs);
00216 int readpcv(const char *file, pcvs_t *pcvs);
00217 pcv_t *searchpcv(int sat, const char *type, gtime_t time,
00218     const pcvs_t *pcvs);
00219 void readpos(const char *file, const char *rcv, double *pos);
00220 int readblqrecord(FILE *fp, double *odisp);
00221 int readblq(const char *file, const char *sta, double *odisp);
00222 int readerp(const char *file, erp_t *erp);
00223 int geterp(const erp_t *erp, gtime_t time, double *erpv);
00224 int cmpeph(const void *p1, const void *p2);
00225 void uniqeph(nav_t *nav);
00226 int cmpgeph(const void *p1, const void *p2);
00227 void uniqgeph(nav_t *nav);
00228 int cmpseph(const void *p1, const void *p2);
00229 void uniqseph(nav_t *nav);
00230 void uniqnav(nav_t *nav);
```

```
00231 int cmpobs(const void *p1, const void *p2);
00232 int sortobs(obs_t *obs);
00233 int screent(gtime_t time, gtime_t ts, gtime_t te, double tint);
00234 int readnav(const char *file, nav_t *nav);
00235 int savenav(const char *file, const nav_t *nav);
00236 void freeobs(obs_t *obs);
00237 void freenav(nav_t *nav, int opt);
00238
00239 void traceopen(const char *file);
00240 void traceclose();
00241 void tracelevel(int level);
00242 void traceswap();
00243 void trace(int level, const char *format, ...);
00244 void tracet(int level, const char *format, ...);
00245 void tracemat(int level, const double *A, int n, int m, int p, int q);
00246 void traceobs(int level, const obsd_t *obs, int n);
00247 // void tracenav(int level, const nav_t *nav);
00248 // void tracegnav(int level, const nav_t *nav);
00249 // void tracehnav(int level, const nav_t *nav);
00250 // void tracepeph(int level, const nav_t *nav);
00251 // void tracepclk(int level, const nav_t *nav);
00252 // void traceb  (int level, const unsigned char *p, int n);
00253
00254 int execcmd(const char *cmd);
00255 void createdir(fs::path const &path);
00256 int reppath(std::string const &path, std::string &rpath, gtime_t time, const char *rov,
00257     const char *base);
00258 double satwavelen(int sat, int frq, const nav_t *nav);
00259 double geodist(const double *rs, const double *rr, double *e);
00260 double satazel(const double *pos, const double *e, double *azel);
00261
00262 void dops(int ns, const double *azel, double elmin, double *dop);
00263 double ionmodel(gtime_t t, const double *ion, const double *pos,
00264     const double *azel);
00265 double ionmapf(const double *pos, const double *azel);
00266 double ionppp(const double *pos, const double *azel, double re,
00267     double hion, double *posp);
00268 double tropmodel(gtime_t time, const double *pos, const double *azel,
00269     double humi);
00270 double interpc(const double coef[], double lat);
00271 double mapf(double el, double a, double b, double c);
00272 double nmf(gtime_t time, const double pos[], const double azel[],
00273     double *mapfw);
00274 double tropmapf(gtime_t time, const double pos[], const double azel[],
00275     double *mapfw);
00276 double interpvar(double ang, const double *var);
00277
00278 void antmodel(const pcv_t *pcv, const double *del, const double *azel,
00279     int opt, double *dant);
00280
00281 void antmodel_s(const pcv_t *pcv, double nadir, double *dant);
00282 void sunmoonpos_eci(gtime_t tut, double *rsun, double *rmoon);
00283 void sunmoonpos(gtime_t tutc, const double *erpv, double *rsun,
00284     double *rmoon, double *gmst);
00285 void csmooth(obs_t *obs, int ns);
00286 int rtk_uncompress(const char *file, char *uncfile);
00287 int expath(const char *path, char *paths[], int nmax);
00288 void windupcorr(gtime_t time, const double *rs, const double *rr, double *phw);
00289
00290 #endif  // GNSS_SDR_RTKLIB_RTKCMN_H
```

13.185 rtklib_rtkpos.h File Reference

rtklib ppp-related functions

```
#include "rtklib.h"
#include "rtklib_rtkcmn.h"
```

Macros

- #define II_RTK(s, opt)
- #define IT_RTK(r, opt)
- #define IL_RTK(f, opt)
- #define IB_RTK(s, f, opt)

Functions

- int **rtkopenstat** (const char *file, int level)

- void **rtkclosestat** ()
- void **rtkoutstat** (rtk_t *rtk)
- void **swapsolstat** ()
- void **outsolstat** (rtk_t *rtk)
- void **errmsg** (rtk_t *rtk, const char *format,...)
- double **sdobs** (const obsd_t *obs, int i, int j, int f)
- double **gfobs_L1L2** (const obsd_t *obs, int i, int j, const double *lam)
- double **gfobs_L1L5** (const obsd_t *obs, int i, int j, const double *lam)
- double **varerr** (int sat, int sys, double el, double snr_rover, double snr_base, double bl, double dt, int f, const prcopt_t *opt, const obsd_t *obs)
- double **baseline** (const double *ru, const double *rb, double *dr)
- void **initx_rtk** (rtk_t *rtk, double xi, double var, int i)
- int **selsat** (const obsd_t *obs, const double *azel, int nu, int nr, const prcopt_t *opt, int *sat, int *iu, int *ir)
- void **udpos** (rtk_t *rtk, double tt)
- void **udion** (rtk_t *rtk, double tt, double bl, const int *sat, int ns)
- void **udtrop** (rtk_t *rtk, double tt, double bl)
- void **udrcvbias** (rtk_t *rtk, double tt)
- void **detslp_ll** (rtk_t *rtk, const obsd_t *obs, int i, int rcv)
- void **detslp_gf_L1L2** (rtk_t *rtk, const obsd_t *obs, int i, int j, const nav_t *nav)
- void **detslp_gf_L1L5** (rtk_t *rtk, const obsd_t *obs, int i, int j, const nav_t *nav)
- void **detslp_dop** (rtk_t *rtk, const obsd_t *obs, const int *ix, int ns, int rcv, const nav_t *nav)
- void **udbias** (rtk_t *rtk, double tt, const obsd_t *obs, const int *sat, const int *iu, const int *ir, int ns, const nav_t *nav)
- void **udstate** (rtk_t *rtk, const obsd_t *obs, const int *sat, const int *iu, const int *ir, int ns, const nav_t *nav)
- void **zdres_sat** (int base, double r, const obsd_t *obs, const nav_t *nav, const double *azel, const double *dant, const prcopt_t *opt, double *y)
- int **zdres** (int base, const obsd_t *obs, int n, const double *rs, const double *dts, const int *svh, const nav_t *nav, const double *rr, const prcopt_t *opt, int index, double *y, double *e, double *azel)
- int **validobs** (int i, int j, int f, int nf, const double *y)
- void **ddcov** (const int *nb, int n, const double *Ri, const double *Rj, int nv, double *R)
- int **constbl** (rtk_t *rtk, const double *x, const double *P, double *v, double *H, double *Ri, double *Rj, int index)
- double **prectrop** (gtime_t time, const double *pos, int r, const double *azel, const prcopt_t *opt, const double *x, double *dtdx)
- double **gloicbcorr** (int sat1, int sat2, const prcopt_t *opt, double lam1, double lam2, int f)
- int **test_sys** (int sys, int m)
- int **ddres** (rtk_t *rtk, const obsd_t *obs, const nav_t *nav, double dt, const double *x, const double *P, const int *sat, double *y, const double *e, double *azel, const int *iu, const int *ir, int ns, double *v, double *H, double *R, int *vflg)
- double **intpres** (gtime_t time, const obsd_t *obs, int n, const nav_t *nav, rtk_t *rtk, double *y)
- int **ddidx** (rtk_t *rtk, int *ix, int glo, int sbs)
- void **restamb** (rtk_t *rtk, const double *bias, int nb, double *xa)
- void **holdamb** (rtk_t *rtk, const double *xa)
- int **resamb_LAMBDA** (rtk_t *rtk, double *bias, double *xa, int glo, int sbs)
- int **manage_amb_LAMBDA** (rtk_t *rtk, double *bias, double *xa, const int *sat, int nf, int ns)
- int **valpos** (rtk_t *rtk, const double *v, const double *R, const int *vflg, int nv, double thres)
- int **relpos** (rtk_t *rtk, const obsd_t *obs, int nu, int nr, const nav_t *nav)
- void **rtkinit** (rtk_t *rtk, const prcopt_t *opt)
- void **rtkfree** (rtk_t *rtk)
- int **rtkpos** (rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav)

Variables

- const double VAR_POS = std::pow(30.0, 2.0)
- const double VAR_VEL = std::pow(10.0, 2.0)
- const double VAR_ACC = std::pow(10.0, 2.0)
- const double VAR_HWBIAS = std::pow(1.0, 2.0)
- const double VAR_GRA = std::pow(0.001, 2.0)
- const double INIT_ZWD = 0.15
- const double PRN_HWBIA = 1E-6
- const double MAXAC = 30.0
- const double VAR_HOLDAMB = 0.001
- const double TTOL_MOVEB = (1.0 + 2 ∗ DTTOL)

13.185.1 Detailed Description

rtklib ppp-related functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

13.185.1.1 SPDX-License-Identifier: BSD-2-Clause

Definition in file rtklib_rtkpos.h.

13.186 rtklib_rtkpos.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtkpos.h
00003  * \brief rtklib ppp-related functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  * ----------------------------------------------------------------------------
```

```
00030  */
00031
00032 #ifndef GNSS_SDR_RTKLIB_RKTPOS_H
00033 #define GNSS_SDR_RTKLIB_RKTPOS_H
00034
00035 #include "rtklib.h"
00036 #include "rtklib_rtkcmn.h"
00037
00038 /** \addtogroup PVT
00039  * \{ */
00040 /** \addtogroup RTKLIB_Library
00041  * \{ */
00042
00043
00044 /* constants/macros ----------------------------------------------------------*/
00045 const double VAR_POS = std::pow(30.0, 2.0);   /* initial variance of receiver pos (m^2) */
00046 const double VAR_VEL = std::pow(10.0, 2.0);   /* initial variance of receiver vel ((m/s)^2) */
00047 const double VAR_ACC = std::pow(10.0, 2.0);   /* initial variance of receiver acc ((m/ss)^2) */
00048 const double VAR_HWBIAS = std::pow(1.0, 2.0); /* initial variance of h/w bias ((m/MHz)^2) */
00049 const double VAR_GRA = std::pow(0.001, 2.0);  /* initial variance of gradient (m^2) */
00050 const double INIT_ZWD = 0.15;                 /* initial zwd (m) */
00051
00052 const double PRN_HWBIA = 1E-6; /* process noise of h/w bias (m/MHz/sqrt(s)) */
00053 const double MAXAC = 30.0;     /* max accel for doppler slip detection (m/s^2) */
00054
00055 const double VAR_HOLDAMB = 0.001; /* constraint to hold ambiguity (cycle^2) */
00056
00057 const double TTOL_MOVEB = (1.0 + 2 * DTTOL);
00058 /* time sync tolerance for moving-baseline (s) */
00059
00060 /* number of parameters (pos,ionos,tropos,hw-bias,phase-bias,real,estimated) */
00061
00062
00063 /* state variable index */
00064 // clang-format off
00065 #define II_RTK(s, opt) (NP_RTK(opt) + (s)-1)                              /* ionos (s:satellite no)
     */
00066 #define IT_RTK(r, opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) / 2 * (r)) /* tropos (r:0=rov,1:ref)
     */
00067 #define IL_RTK(f, opt) (NP_RTK(opt) + NI_RTK(opt) + NT_RTK(opt) + (f))     /* receiver h/w bias */
00068 #define IB_RTK(s, f, opt) (NR_RTK(opt) + MAXSAT * (f) + (s)-1)             /* phase bias
     (s:satno,f:freq) */
00069 // clang-format on
00070
00071 int rtkopenstat(const char *file, int level);
00072
00073 void rtkclosestat();
00074
00075 void rtkoutstat(rtk_t *rtk);
00076
00077 void swapsolstat();
00078
00079 void outsolstat(rtk_t *rtk);
00080
00081 void errmsg(rtk_t *rtk, const char *format, ...);
00082
00083 double sdobs(const obsd_t *obs, int i, int j, int f);
00084
00085 double gfobs_L1L2(const obsd_t *obs, int i, int j, const double *lam);
00086
00087 double gfobs_L1L5(const obsd_t *obs, int i, int j, const double *lam);
00088
00089 double varerr(int sat, int sys, double el, double snr_rover, double snr_base,
00090     double bl, double dt, int f, const prcopt_t *opt, const obsd_t *obs);
00091
00092
00093 double baseline(const double *ru, const double *rb, double *dr);
00094
00095 void initx_rtk(rtk_t *rtk, double xi, double var, int i);
00096
00097 int selsat(const obsd_t *obs, const double *azel, int nu, int nr,
00098     const prcopt_t *opt, int *sat, int *iu, int *ir);
00099
00100 void udpos(rtk_t *rtk, double tt);
00101
00102 void udion(rtk_t *rtk, double tt, double bl, const int *sat, int ns);
00103
00104 void udtrop(rtk_t *rtk, double tt, double bl);
00105
00106 void udrcvbias(rtk_t *rtk, double tt);
00107
00108 void detslp_ll(rtk_t *rtk, const obsd_t *obs, int i, int rcv);
00109 void detslp_gf_L1L2(rtk_t *rtk, const obsd_t *obs, int i, int j,
00110     const nav_t *nav);
00111
00112 void detslp_gf_L1L5(rtk_t *rtk, const obsd_t *obs, int i, int j,
00113     const nav_t *nav);
```

```
00114
00115 void detslp_dop(rtk_t *rtk, const obsd_t *obs, const int *ix, int ns, int rcv,
00116     const nav_t *nav);
00117
00118 void udbias(rtk_t *rtk, double tt, const obsd_t *obs, const int *sat,
00119     const int *iu, const int *ir, int ns, const nav_t *nav);
00120
00121 void udstate(rtk_t *rtk, const obsd_t *obs, const int *sat,
00122     const int *iu, const int *ir, int ns, const nav_t *nav);
00123
00124 void zdres_sat(int base, double r, const obsd_t *obs, const nav_t *nav,
00125     const double *azel, const double *dant,
00126     const prcopt_t *opt, double *y);
00127
00128 int zdres(int base, const obsd_t *obs, int n, const double *rs,
00129     const double *dts, const int *svh, const nav_t *nav,
00130     const double *rr, const prcopt_t *opt, int index, double *y,
00131     double *e, double *azel);
00132
00133 int validobs(int i, int j, int f, int nf, const double *y);
00134
00135 void ddcov(const int *nb, int n, const double *Ri, const double *Rj,
00136     int nv, double *R);
00137
00138 int constbl(rtk_t *rtk, const double *x, const double *P, double *v,
00139     double *H, double *Ri, double *Rj, int index);
00140
00141 double prectrop(gtime_t time, const double *pos, int r,
00142     const double *azel, const prcopt_t *opt, const double *x,
00143     double *dtdx);
00144
00145 double gloicbcorr(int sat1, int sat2, const prcopt_t *opt, double lam1,
00146     double lam2, int f);
00147
00148 int test_sys(int sys, int m);
00149
00150 int ddres(rtk_t *rtk, const obsd_t *obs, const nav_t *nav, double dt, const double *x,
00151     const double *P, const int *sat, double *y, const double *e,
00152     double *azel, const int *iu, const int *ir, int ns, double *v,
00153     double *H, double *R, int *vflg);
00154
00155 double intpres(gtime_t time, const obsd_t *obs, int n, const nav_t *nav,
00156     rtk_t *rtk, double *y);
00157
00158
00159 int ddidx(rtk_t *rtk, int *ix, int glo, int sbs);
00160
00161 void restamb(rtk_t *rtk, const double *bias, int nb, double *xa);
00162
00163 void holdamb(rtk_t *rtk, const double *xa);
00164
00165 int resamb_LAMBDA(rtk_t *rtk, double *bias, double *xa, int glo, int sbs);
00166
00167 int manage_amb_LAMBDA(rtk_t *rtk, double *bias, double *xa, const int *sat,
00168     int nf, int ns);
00169
00170 int valpos(rtk_t *rtk, const double *v, const double *R, const int *vflg,
00171     int nv, double thres);
00172
00173 int relpos(rtk_t *rtk, const obsd_t *obs, int nu, int nr,
00174     const nav_t *nav);
00175
00176 void rtkinit(rtk_t *rtk, const prcopt_t *opt);
00177
00178 void rtkfree(rtk_t *rtk);
00179
00180 int rtkpos(rtk_t *rtk, const obsd_t *obs, int n, const nav_t *nav);
00181
00182
00183 /** \} */
00184 /** \} */
00185 #endif  // GNSS_SDR_RTKLIB_RKTPOS_H
```

13.187 rtklib_rtksvr.h File Reference

rtk server functions

`#include "rtklib.h"`

Functions

- void **writesolhead** (stream_t *stream, const solopt_t *solopt)
- void **saveoutbuf** (rtksvr_t *svr, unsigned char *buff, int n, int index)
- void **writesol** (rtksvr_t *svr, int index)
- void **updatenav** (nav_t *nav)
- void **updatefcn** (rtksvr_t *svr)
- void **updatesvr** (rtksvr_t *svr, int ret, obs_t *obs, nav_t *nav, int sat, sbsmsg_t *sbsmsg, int index, int iobs)
- int **decoderaw** (rtksvr_t *svr, int index)
- void **decodefile** (rtksvr_t *svr, int index)
- void * **rtksvrthread** (void *arg)
- int **rtksvrinit** (rtksvr_t *svr)
- void **rtksvrfree** (rtksvr_t *svr)
- void **rtksvrlock** (rtksvr_t *svr)
- void **rtksvrunlock** (rtksvr_t *svr)
- int **rtksvrstart** (rtksvr_t *svr, int cycle, int buffsize, int *strs, char **paths, const int *formats, int navsel, char **cmds, char **rcvopts, int nmeacycle, int nmeareq, const double *nmeapos, prcopt_t *prcopt, solopt_t *solopt, stream_t *moni)
- void **rtksvrstop** (rtksvr_t *svr, char **cmds)
- int **rtksvropenstr** (rtksvr_t *svr, int index, int str, const char *path, const solopt_t *solopt)
- void **rtksvrclosestr** (rtksvr_t *svr, int index)
- int **rtksvrostat** (rtksvr_t *svr, int rcv, gtime_t *time, int *sat, double *az, double *el, int **snr, int *vsat)
- void **rtksvrsstat** (rtksvr_t *svr, int *sstat, char *msg)

Variables

- const solopt_t SOLOPT_DEFAULT
- const prcopt_t PRCOPT_DEFAULT

13.187.1 Detailed Description

rtk server functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_rtksvr.h.

13.187.2 Variable Documentation

13.187.2.1 PRCOPT_DEFAULT

```
const prcopt_t PRCOPT_DEFAULT
```

Initial value:

```
                        = {
PMODE_SINGLE, 0, 2, SYS_GPS,
15.0 * D2R, {{}, {{}, {}}},
0, 1, 1, 1,
5, 0, 10, 1,
```

```
0, 0, 0, 0,
1, 0, 0, 0, 0,
0, 0,
{100.0, 100.0, 100.0},
{100.0, 0.003, 0.003, 0.0, 1.0},
{30.0, 0.03, 0.3},
{1e-4, 1e-3, 1e-4, 1e-1, 1e-2, 0.0},
5E-12,
{3.0, 0.9999, 0.25, 0.1, 0.05, 0, 0, 0},
0.0, 0.0, 0.05, 0.0,
30.0, {30.0, 30.0}, 30.0,
{}, {}, {},
{"", ""},
{}, {}, {},
0, 0, 0, {"", ""}, {}, 0, {{}, {}}, {{}, {{}, {}}, {{}, {}}, {}, {}}, 0, {}, true,
false,
0, 0, 0, 0, 0.0, 0.0}
```

Definition at line 49 of file rtklib_rtksvr.h.

13.187.2.2 SOLOPT_DEFAULT

```
const solopt_t SOLOPT_DEFAULT
```

Initial value:

```
                              = {

    SOLF_LLH, TIMES_GPST, 1, 3,
    0, 1, 0, 0, 0, 0,
    0, 0, 0,
    {0.0, 0.0},
    " ", "", 0
}
```

Definition at line 40 of file rtklib_rtksvr.h.

13.188 rtklib_rtksvr.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_rtksvr.h
00003  * \brief rtk server functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * ----------------------------------------------------------------------------
00032  */
00033
00034 #ifndef GNSS_SDR_RTKLIB_RKTSVR_H
00035 #define GNSS_SDR_RTKLIB_RKTSVR_H
00036
00037 #include "rtklib.h"
00038
00039
00040 const solopt_t SOLOPT_DEFAULT = {
00041     /* defaults solution output options */
00042     SOLF_LLH, TIMES_GPST, 1, 3, /* posf, times, timef, timeu */
00043     0, 1, 0, 0, 0, 0,           /* degf, outhead, outopt, datum, height, geoid */
00044     0, 0, 0,                    /* solstatic, sstat, trace */
```

```
00045     {0.0, 0.0},                 /* nmeaintv */
00046     " ", "", 0                  /* separator/program name */
00047 };
00048
00049 const prcopt_t PRCOPT_DEFAULT = {           /* defaults processing options */
00050     PMODE_SINGLE, 0, 2, SYS_GPS,            /* mode, soltype, nf, navsys */
00051     15.0 * D2R, {{}, {{}, {}}},             /* elmin, snrmask */
00052     0, 1, 1, 1,                             /* sateph, modear, glomodear, bdsmodear */
00053     5, 0, 10, 1,                            /* maxout, minlock, minfix, armaxiter */
00054     0, 0, 0, 0,                             /* estion, esttrop, dynamics, tidecorr */
00055     1, 0, 0, 0, 0,                          /* niter, codesmooth, intpref, sbascorr, sbassatsel */
00056     0, 0,                                   /* rovpos, refpos */
00057     {100.0, 100.0, 100.0},                  /* eratio[] */
00058     {100.0, 0.003, 0.003, 0.0, 1.0},        /* err[] */
00059     {30.0, 0.03, 0.3},                      /* std[] */
00060     {1e-4, 1e-3, 1e-4, 1e-1, 1e-2, 0.0},    /* prn[] */
00061     5E-12,                                  /* sclkstab */
00062     {3.0, 0.9999, 0.25, 0.1, 0.05, 0, 0, 0}, /* thresar */
00063     0.0, 0.0, 0.05, 0.0,                    /* elmaskar, almaskhold, thresslip, thresdop */
00064     30.0, {30.0, 30.0}, 30.0,               /* maxtdif, maxinno{phase,code}, maxgdop */
00065     {}, {}, {},                             /* baseline, ru, rb */
00066     {"", ""},                               /* anttype */
00067     {}, {}, {},                             /* antdel, pcv, exsats */
00068     0, 0, 0, {"", ""}, {}, 0, {{}, {}}, {{}, {{}, {}}, {{}, {}}, {}, {}}, 0, {}, true,
00069     false,                  /* estqzsisb */
00070     0, 0, 0, 0, 0.0, 0.0}; /* arfilter, minfixsats, minholdsats, mindropsats, varholdamb, armaxposvar
      */
00071
00072
00073 void writesolhead(stream_t *stream, const solopt_t *solopt);
00074
00075 void saveoutbuf(rtksvr_t *svr, unsigned char *buff, int n, int index);
00076
00077 void writesol(rtksvr_t *svr, int index);
00078
00079 void updatenav(nav_t *nav);
00080
00081 void updatefcn(rtksvr_t *svr);
00082
00083 void updatesvr(rtksvr_t *svr, int ret, obs_t *obs, nav_t *nav, int sat,
00084     sbsmsg_t *sbsmsg, int index, int iobs);
00085
00086 int decoderaw(rtksvr_t *svr, int index);
00087
00088 void decodefile(rtksvr_t *svr, int index);
00089
00090 void *rtksvrthread(void *arg);
00091
00092 int rtksvrinit(rtksvr_t *svr);
00093
00094 void rtksvrfree(rtksvr_t *svr);
00095
00096 void rtksvrlock(rtksvr_t *svr);
00097
00098 void rtksvrunlock(rtksvr_t *svr);
00099
00100 int rtksvrstart(rtksvr_t *svr, int cycle, int buffsize, int *strs,
00101     char **paths, const int *formats, int navsel, char **cmds,
00102     char **rcvopts, int nmeacycle, int nmeareq,
00103     const double *nmeapos, prcopt_t *prcopt,
00104     solopt_t *solopt, stream_t *moni);
00105
00106 void rtksvrstop(rtksvr_t *svr, char **cmds);
00107
00108 int rtksvropenstr(rtksvr_t *svr, int index, int str, const char *path,
00109     const solopt_t *solopt);
00110
00111 void rtksvrclosestr(rtksvr_t *svr, int index);
00112
00113 int rtksvrostat(rtksvr_t *svr, int rcv, gtime_t *time, int *sat,
00114     double *az, double *el, int **snr, int *vsat);
00115
00116 void rtksvrsstat(rtksvr_t *svr, int *sstat, char *msg);
00117
00118
00119 #endif
```

13.189 rtklib_sbas.h File Reference

sbas functions

```
#include "rtklib.h"
```

Functions

- char * **getfield** (char *p, int pos)
- double **varfcorr** (int udre)
- double **varicorr** (int give)
- double **degfcorr** (int ai)
- int **decode_sbstype1** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype2** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype6** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype7** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype9** (const sbsmsg_t *msg, nav_t *nav)
- int **decode_sbstype18** (const sbsmsg_t *msg, sbsion_t *sbsion)
- int **decode_longcorr0** (const sbsmsg_t *msg, int p, sbssat_t *sbssat)
- int **decode_longcorr1** (const sbsmsg_t *msg, int p, sbssat_t *sbssat)
- int **decode_longcorrh** (const sbsmsg_t *msg, int p, sbssat_t *sbssat)
- int **decode_sbstype24** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype25** (const sbsmsg_t *msg, sbssat_t *sbssat)
- int **decode_sbstype26** (const sbsmsg_t *msg, sbsion_t *sbsion)
- int **sbsupdatecorr** (const sbsmsg_t *msg, nav_t *nav)
- void **readmsgs** (const char *file, int sel, gtime_t ts, gtime_t te, sbs_t *sbs)
- int **cmpmsgs** (const void *p1, const void *p2)
- int **sbsreadmsgt** (const char *file, int sel, gtime_t ts, gtime_t te, sbs_t *sbs)
- int **sbsreadmsg** (const char *file, int sel, sbs_t *sbs)
- void **sbsoutmsg** (FILE *fp, sbsmsg_t *sbsmsg)
- void **searchigp** (gtime_t time, const double *pos, const sbsion_t *ion, const sbsigp_t **igp, double *x, double *y)
- int **sbsioncorr** (gtime_t time, const nav_t *nav, const double *pos, const double *azel, double *delay, double *var)
- void **getmet** (double lat, double *met)
- double **sbstropcorr** (gtime_t time, const double *pos, const double *azel, double *var)
- int **sbslongcorr** (gtime_t time, int sat, const sbssat_t *sbssat, double *drs, double *ddts)
- int **sbsfastcorr** (gtime_t time, int sat, const sbssat_t *sbssat, double *prc, double *var)
- int **sbssatcorr** (gtime_t time, int sat, const nav_t *nav, double *rs, double *dts, double *var)
- int **sbsdecodemsg** (gtime_t time, int prn, const unsigned int *words, sbsmsg_t *sbsmsg)

Variables

- const int WEEKOFFSET = 1024
- const sbsigpband_t IGPBAND1 [9][8]
- const sbsigpband_t IGPBAND2 [2][5]

13.189.1 Detailed Description

sbas functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References : [1] RTCA/DO-229C, Minimum operational performance standards for global positioning system/wide area augmentation system airborne equipment, RTCA inc, November 28, 2001 [2] IS-QZSS v.1.1, Quasi-Zenith Satellite System Navigation Service Interface Specification for QZSS, Japan Aerospace Exploration Agency, July 31, 2009
Definition in file rtklib_sbas.h.

13.189.2 Variable Documentation

13.189.2.1 IGPBAND1

```
const sbsigpband_t IGPBAND1[9][8]
```

Initial value:

```
                    = {
{{-180, X1, 1, 28}, {-175, X2, 29, 51}, {-170, X3, 52, 78}, {-165, X2, 79, 101},
    {-160, X3, 102, 128}, {-155, X2, 129, 151}, {-150, X3, 152, 178}, {-145, X2, 179, 201}},
{{-140, X4, 1, 28}, {-135, X2, 29, 51}, {-130, X3, 52, 78}, {-125, X2, 79, 101},
    {-120, X3, 102, 128}, {-115, X2, 129, 151}, {-110, X3, 152, 178}, {-105, X2, 179, 201}},
{{-100, X3, 1, 27}, {-95, X2, 28, 50}, {-90, X1, 51, 78}, {-85, X2, 79, 101},
    {-80, X3, 102, 128}, {-75, X2, 129, 151}, {-70, X3, 152, 178}, {-65, X2, 179, 201}},
{{-60, X3, 1, 27}, {-55, X2, 28, 50}, {-50, X4, 51, 78}, {-45, X2, 79, 101},
    {-40, X3, 102, 128}, {-35, X2, 129, 151}, {-30, X3, 152, 178}, {-25, X2, 179, 201}},
{{-20, X3, 1, 27}, {-15, X2, 28, 50}, {-10, X3, 51, 77}, {-5, X2, 78, 100},
    {0, X1, 101, 128}, {5, X2, 129, 151}, {10, X3, 152, 178}, {15, X2, 179, 201}},
{{20, X3, 1, 27}, {25, X2, 28, 50}, {30, X3, 51, 77}, {35, X2, 78, 100},
    {40, X4, 101, 128}, {45, X2, 129, 151}, {50, X3, 152, 178}, {55, X2, 179, 201}},
{{60, X3, 1, 27}, {65, X2, 28, 50}, {70, X3, 51, 77}, {75, X2, 78, 100},
    {80, X3, 101, 127}, {85, X2, 128, 150}, {90, X1, 151, 178}, {95, X2, 179, 201}},
{{100, X3, 1, 27}, {105, X2, 28, 50}, {110, X3, 51, 77}, {115, X2, 78, 100},
    {120, X3, 101, 127}, {125, X2, 128, 150}, {130, X4, 151, 178}, {135, X2, 179, 201}},
{{140, X3, 1, 27}, {145, X2, 28, 50}, {150, X3, 51, 77}, {155, X2, 78, 100},
    {160, X3, 101, 127}, {165, X2, 128, 150}, {170, X3, 151, 177}, {175, X2, 178, 200}}}
```

Definition at line 73 of file rtklib_sbas.h.

13.189.2.2 IGPBAND2

```
const sbsigpband_t IGPBAND2[2][5]
```

Initial value:

```
                    = {
{{60, X5, 1, 72}, {65, X6, 73, 108}, {70, X6, 109, 144}, {75, X6, 145, 180},
    {85, X7, 181, 192}},
{{-60, X5, 1, 72}, {-65, X6, 73, 108}, {-70, X6, 109, 144}, {-75, X6, 145, 180},
    {-85, X8, 181, 192}}}
```

Definition at line 92 of file rtklib_sbas.h.

13.189.2.3 WEEKOFFSET

```
const int WEEKOFFSET = 1024
```

Definition at line 49 of file rtklib_sbas.h.

13.190 rtklib_sbas.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_sbas.h
00003  * \brief sbas functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
```

```
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * -----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *
00031  * References :
00032  *     [1] RTCA/DO-229C, Minimum operational performance standards for global
00033  *         positioning system/wide area augmentation system airborne equipment,
00034  *         RTCA inc, November 28, 2001
00035  *     [2] IS-QZSS v.1.1, Quasi-Zenith Satellite System Navigation Service
00036  *         Interface Specification for QZSS, Japan Aerospace Exploration Agency,
00037  *         July 31, 2009
00038  *
00039  * -----------------------------------------------------------------------------
00040  */
00041
00042 #ifndef GNSS_SDR_RTKLIB_SBAS_H
00043 #define GNSS_SDR_RTKLIB_SBAS_H
00044
00045 #include "rtklib.h"
00046
00047 /* constants -----------------------------------------------------------------*/
00048
00049 const int WEEKOFFSET = 1024; /* gps week offset for NovAtel OEM-3 */
00050
00051 /* sbas igp definition -------------------------------------------------------*/
00052 static const short
00053     X1[] = {-75, -65, -55, -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20,
00054         25, 30, 35, 40, 45, 50, 55, 65, 75, 85},
00055     X2[] = {-55, -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20, 25, 30,
00056         35, 40, 45, 50, 55},
00057     X3[] = {-75, -65, -55, -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20,
00058         25, 30, 35, 40, 45, 50, 55, 65, 75},
00059     X4[] = {-85, -75, -65, -55, -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, 5, 10, 15,
00060         20, 25, 30, 35, 40, 45, 50, 55, 65, 75},
00061     X5[] = {-180, -175, -170, -165, -160, -155, -150, -145, -140, -135, -130, -125, -120, -115,
00062         -110, -105, -100, -95, -90, -85, -80, -75, -70, -65, -60, -55, -50, -45,
00063         -40, -35, -30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20, 25,
00064         30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95,
00065         100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165,
00066         170, 175},
00067     X6[] = {-180, -170, -160, -150, -140, -130, -120, -110, -100, -90, -80, -70, -60, -50,
00068         -40, -30, -20, -10, 0, 10, 20, 30, 40, 50, 60, 70, 80, 90,
00069         100, 110, 120, 130, 140, 150, 160, 170},
00070     X7[] = {-180, -150, -120, -90, -60, -30, 0, 30, 60, 90, 120, 150},
00071     X8[] = {-170, -140, -110, -80, -50, -20, 10, 40, 70, 100, 130, 160};
00072
00073 const sbsigpband_t IGPBAND1[9][8] = {/* band 0-8 */
00074     {{-180, X1, 1, 28}, {-175, X2, 29, 51}, {-170, X3, 52, 78}, {-165, X2, 79, 101},
00075         {-160, X3, 102, 128}, {-155, X2, 129, 151}, {-150, X3, 152, 178}, {-145, X2, 179, 201}},
00076     {{-140, X4, 1, 28}, {-135, X2, 29, 51}, {-130, X3, 52, 78}, {-125, X2, 79, 101},
00077         {-120, X3, 102, 128}, {-115, X2, 129, 151}, {-110, X3, 152, 178}, {-105, X2, 179, 201}},
00078     {{-100, X3, 1, 27}, {-95, X2, 28, 50}, {-90, X1, 51, 78}, {-85, X2, 79, 101},
00079         {-80, X3, 102, 128}, {-75, X2, 129, 151}, {-70, X3, 152, 178}, {-65, X2, 179, 201}},
00080     {{-60, X3, 1, 27}, {-55, X2, 28, 50}, {-50, X4, 51, 78}, {-45, X2, 79, 101},
00081         {-40, X3, 102, 128}, {-35, X2, 129, 151}, {-30, X3, 152, 178}, {-25, X2, 179, 201}},
00082     {{-20, X3, 1, 27}, {-15, X2, 28, 50}, {-10, X3, 51, 77}, {-5, X2, 78, 100},
00083         {0, X1, 101, 128}, {5, X2, 129, 151}, {10, X3, 152, 178}, {15, X2, 179, 201}},
00084     {{20, X3, 1, 27}, {25, X2, 28, 50}, {30, X3, 51, 77}, {35, X2, 78, 100},
00085         {40, X4, 101, 128}, {45, X2, 129, 151}, {50, X3, 152, 178}, {55, X2, 179, 201}},
00086     {{60, X3, 1, 27}, {65, X2, 28, 50}, {70, X3, 51, 77}, {75, X2, 78, 100},
00087         {80, X3, 101, 127}, {85, X2, 128, 150}, {90, X1, 151, 178}, {95, X2, 179, 201}},
00088     {{100, X3, 1, 27}, {105, X2, 28, 50}, {110, X3, 51, 77}, {115, X2, 78, 100},
00089         {120, X3, 101, 127}, {125, X2, 128, 150}, {130, X4, 151, 178}, {135, X2, 179, 201}},
00090     {{140, X3, 1, 27}, {145, X2, 28, 50}, {150, X3, 51, 77}, {155, X2, 78, 100},
00091         {160, X3, 101, 127}, {165, X2, 128, 150}, {170, X3, 151, 177}, {175, X2, 178, 200}}};
00092 const sbsigpband_t IGPBAND2[2][5] = {/* band 9-10 */
00093     {{60, X5, 1, 72}, {65, X6, 73, 108}, {70, X6, 109, 144}, {75, X6, 145, 180},
00094         {85, X7, 181, 192}},
00095     {{-60, X5, 1, 72}, {-65, X6, 73, 108}, {-70, X6, 109, 144}, {-75, X6, 145, 180},
00096         {-85, X8, 181, 192}}};
00097
00098
00099 char *getfield(char *p, int pos);
00100 double varfcorr(int udre);
00101 double varicorr(int give);
00102 double degfcorr(int ai);
00103
```

```
00104 int decode_sbstype1(const sbsmsg_t *msg, sbssat_t *sbssat);
00105 int decode_sbstype2(const sbsmsg_t *msg, sbssat_t *sbssat);
00106 int decode_sbstype6(const sbsmsg_t *msg, sbssat_t *sbssat);
00107 int decode_sbstype7(const sbsmsg_t *msg, sbssat_t *sbssat);
00108 int decode_sbstype9(const sbsmsg_t *msg, nav_t *nav);
00109 int decode_sbstype18(const sbsmsg_t *msg, sbsion_t *sbsion);
00110 int decode_longcorr0(const sbsmsg_t *msg, int p, sbssat_t *sbssat);
00111 int decode_longcorr1(const sbsmsg_t *msg, int p, sbssat_t *sbssat);
00112 int decode_longcorrh(const sbsmsg_t *msg, int p, sbssat_t *sbssat);
00113 int decode_sbstype24(const sbsmsg_t *msg, sbssat_t *sbssat);
00114 int decode_sbstype25(const sbsmsg_t *msg, sbssat_t *sbssat);
00115 int decode_sbstype26(const sbsmsg_t *msg, sbsion_t *sbsion);
00116
00117 int sbsupdatecorr(const sbsmsg_t *msg, nav_t *nav);
00118 void readmsgs(const char *file, int sel, gtime_t ts, gtime_t te, sbs_t *sbs);
00119 int cmpmsgs(const void *p1, const void *p2);
00120 int sbsreadmsgt(const char *file, int sel, gtime_t ts, gtime_t te,
00121     sbs_t *sbs);
00122 int sbsreadmsg(const char *file, int sel, sbs_t *sbs);
00123 void sbsoutmsg(FILE *fp, sbsmsg_t *sbsmsg);
00124 void searchigp(gtime_t time, const double *pos, const sbsion_t *ion,
00125     const sbsigp_t **igp, double *x, double *y);
00126 int sbsioncorr(gtime_t time, const nav_t *nav, const double *pos,
00127     const double *azel, double *delay, double *var);
00128
00129 void getmet(double lat, double *met);
00130 double sbstropcorr(gtime_t time, const double *pos, const double *azel,
00131     double *var);
00132 int sbslongcorr(gtime_t time, int sat, const sbssat_t *sbssat,
00133     double *drs, double *ddts);
00134 int sbsfastcorr(gtime_t time, int sat, const sbssat_t *sbssat,
00135     double *prc, double *var);
00136
00137 int sbssatcorr(gtime_t time, int sat, const nav_t *nav, double *rs,
00138     double *dts, double *var);
00139 int sbsdecodemsg(gtime_t time, int prn, const unsigned int *words,
00140     sbsmsg_t *sbsmsg);
00141
00142
00143 #endif  // GNSS_SDR_RTKLIB_SBAS_H
```

13.191 rtklib_secure_wipe.h File Reference

Single declaration of the credential-memory wipe primitive.

`#include <cstddef>`

Functions

- void **secure_wipe** (void *data, std::size_t size)

13.191.1 Detailed Description

Single declaration of the credential-memory wipe primitive.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_secure_wipe.h.

13.192 rtklib_secure_wipe.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_secure_wipe.h
00003  * \brief Single declaration of the credential-memory wipe primitive
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00009  * This file is part of GNSS-SDR.
00010  *
00011  * SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: BSD-2-Clause
00013  *
00014  * ------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_RTKLIB_SECURE_WIPE_H
00018 #define GNSS_SDR_RTKLIB_SECURE_WIPE_H
00019
00020 #include <cstddef>
00021
00022 /** \addtogroup PVT
00023  * \{ */
00024 /** \addtogroup RTKLIB
00025  * \{ */
00026
00027 /* Overwrites credential-bearing memory with volatile stores so the wipe is
00028  * not optimized away (a memset before free is a legal elision target).
00029  * Single wipe primitive behind every credential scrub in the code base,
00030  * defined in rtklib_stream.cc. This header carries the one declaration so
00031  * light-weight consumers (secure_string.h) need not pull the heavy rtklib
00032  * headers. */
00033 void secure_wipe(void *data, std::size_t size);
00034
00035 /** \} */
00036 /** \} */
00037 #endif  // GNSS_SDR_RTKLIB_SECURE_WIPE_H
```

13.193 rtklib_solution.h File Reference

solution functions headers

`#include "rtklib.h"`

Macros

- #define COMMENTH "%"
- #define MSG_DISCONN "$_DISCONNECT\r\n"

Functions

- const char * **opt2sep** (const solopt_t *opt)
- int **tonum** (char *buff, const char *sep, double *v)
- double **sqvar** (double covar)
- double **dmm2deg** (double dmm)
- void **septime** (double t, double *t1, double *t2, double *t3)
- void **soltocov** (const sol_t *sol, double *P)
- void **covtosol** (const double *P, sol_t *sol)
- int **decode_nmearmc** (char **val, int n, sol_t *sol)
- int **decode_nmeagga** (char **val, int n, sol_t *sol)
- int **decode_nmea** (char *buff, sol_t *sol)
- char * **decode_soltime** (char *buff, const solopt_t *opt, gtime_t *time)
- int **decode_solxyz** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solllh** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solenu** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solgsi** (char *buff, const solopt_t *opt, sol_t *sol)
- int **decode_solpos** (char *buff, const solopt_t *opt, sol_t *sol)
- void **decode_refpos** (char *buff, const solopt_t *opt, double *rb)
- int **decode_sol** (char *buff, const solopt_t *opt, sol_t *sol, double *rb)
- void **decode_solopt** (char *buff, solopt_t *opt)
- void **readsolopt** (FILE *fp, solopt_t *opt)
- int **inputsol** (unsigned char data, gtime_t ts, gtime_t te, double tint, int qflag, const solopt_t *opt, solbuf_t *solbuf)
- int **readsoldata** (FILE *fp, gtime_t ts, gtime_t te, double tint, int qflag, const solopt_t *opt, solbuf_t *solbuf)

- int **cmpsol** (const void *p1, const void *p2)
- int **sort_solbuf** (solbuf_t *solbuf)
- int **readsolt** (char *files[ ], int nfile, gtime_t ts, gtime_t te, double tint, int qflag, solbuf_t *solbuf)
- int **readsol** (char *files[ ], int nfile, solbuf_t *sol)
- int **addsol** (solbuf_t *solbuf, const sol_t *sol)
- sol_t * **getsol** (solbuf_t *solbuf, int index)
- void **initsolbuf** (solbuf_t *solbuf, int cyclic, int nmax)
- void **freesolbuf** (solbuf_t *solbuf)
- void **freesolstatbuf** (solstatbuf_t *solstatbuf)
- int **cmpsolstat** (const void *p1, const void *p2)
- int **sort_solstat** (solstatbuf_t *statbuf)
- int **decode_solstat** (char *buff, solstat_t *stat)
- void **addsolstat** (solstatbuf_t *statbuf, const solstat_t *stat)
- int **readsolstatdata** (FILE *fp, gtime_t ts, gtime_t te, double tint, solstatbuf_t *statbuf)
- int **readsolstatt** (char *files[ ], int nfile, gtime_t ts, gtime_t te, double tint, solstatbuf_t *statbuf)
- int **readsolstat** (char *files[ ], int nfile, solstatbuf_t *statbuf)
- int **outecef** (unsigned char *buff, const char *s, const sol_t *sol, const solopt_t *opt)
- int **outpos** (unsigned char *buff, const char *s, const sol_t *sol, const solopt_t *opt)
- int **outenu** (unsigned char *buff, const char *s, const sol_t *sol, const double *rb, const solopt_t *opt)
- int **outnmea_rmc** (unsigned char *buff, const sol_t *sol)
- int **outnmea_gga** (unsigned char *buff, const sol_t *sol)
- int **outnmea_gsa** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat)
- int **outnmea_gsv** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat)
- int **outprcopts** (unsigned char *buff, const prcopt_t *opt)
- int **outsolheads** (unsigned char *buff, const solopt_t *opt)
- int **outsols** (unsigned char *buff, const sol_t *sol, const double *rb, const solopt_t *opt)
- int **outsolexs** (unsigned char *buff, const sol_t *sol, const ssat_t *ssat, const solopt_t *opt)
- void **outprcopt** (FILE *fp, const prcopt_t *opt)
- void **outsolhead** (FILE *fp, const solopt_t *opt)
- void **outsol** (FILE *fp, const sol_t *sol, const double *rb, const solopt_t *opt)
- void **outsolex** (FILE *fp, const sol_t *sol, const ssat_t *ssat, const solopt_t *opt)

13.193.1 Detailed Description

solution functions headers

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

13.193.1.1 SPDX-License-Identifier: BSD-2-Clause

Definition in file rtklib_solution.h.

13.194 rtklib_solution.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_solution.h
00003  * \brief solution functions headers
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *------------------------------------------------------------------------------
00030  */
00031
00032 #ifndef GNSS_SDR_RTKLIB_SOLUTION_H
00033 #define GNSS_SDR_RTKLIB_SOLUTION_H
00034
00035 #include "rtklib.h"
00036
00037 /** \addtogroup PVT
00038  * \{ */
00039 /** \addtogroup RTKLIB_Library
00040  * \{ */
00041
00042
00043 #define COMMENTH "%"                    /* comment line indicator for solution */
00044 #define MSG_DISCONN "$_DISCONNECT\r\n" /* disconnect message */
00045
00046 const char *opt2sep(const solopt_t *opt);
00047
00048 int tonum(char *buff, const char *sep, double *v);
00049
00050 double sqvar(double covar);
00051
00052 double dmm2deg(double dmm);
00053
00054 void septime(double t, double *t1, double *t2, double *t3);
00055
00056 void soltocov(const sol_t *sol, double *P);
00057
00058 void covtosol(const double *P, sol_t *sol);
00059
00060 int decode_nmearmc(char **val, int n, sol_t *sol);
00061
00062 int decode_nmeagga(char **val, int n, sol_t *sol);
00063
00064 int decode_nmea(char *buff, sol_t *sol);
00065
00066 char *decode_soltime(char *buff, const solopt_t *opt, gtime_t *time);
00067
00068 int decode_solxyz(char *buff, const solopt_t *opt, sol_t *sol);
00069
00070 int decode_solllh(char *buff, const solopt_t *opt, sol_t *sol);
00071
00072 int decode_solenu(char *buff, const solopt_t *opt, sol_t *sol);
00073
00074 int decode_solgsi(char *buff, const solopt_t *opt, sol_t *sol);
00075
00076 int decode_solpos(char *buff, const solopt_t *opt, sol_t *sol);
00077
00078 void decode_refpos(char *buff, const solopt_t *opt, double *rb);
00079
00080 int decode_sol(char *buff, const solopt_t *opt, sol_t *sol, double *rb);
00081
00082 void decode_solopt(char *buff, solopt_t *opt);
00083
```

```
00084 void readsolopt(FILE *fp, solopt_t *opt);
00085
00086 int inputsol(unsigned char data, gtime_t ts, gtime_t te, double tint,
00087     int qflag, const solopt_t *opt, solbuf_t *solbuf);
00088
00089 int readsoldata(FILE *fp, gtime_t ts, gtime_t te, double tint, int qflag,
00090     const solopt_t *opt, solbuf_t *solbuf);
00091
00092 int cmpsol(const void *p1, const void *p2);
00093
00094 int sort_solbuf(solbuf_t *solbuf);
00095
00096 int readsolt(char *files[], int nfile, gtime_t ts, gtime_t te,
00097     double tint, int qflag, solbuf_t *solbuf);
00098
00099 int readsol(char *files[], int nfile, solbuf_t *sol);
00100
00101 int addsol(solbuf_t *solbuf, const sol_t *sol);
00102
00103 sol_t *getsol(solbuf_t *solbuf, int index);
00104
00105 void initsolbuf(solbuf_t *solbuf, int cyclic, int nmax);
00106
00107 void freesolbuf(solbuf_t *solbuf);
00108
00109 void freesolstatbuf(solstatbuf_t *solstatbuf);
00110
00111 int cmpsolstat(const void *p1, const void *p2);
00112
00113 int sort_solstat(solstatbuf_t *statbuf);
00114
00115 int decode_solstat(char *buff, solstat_t *stat);
00116
00117 void addsolstat(solstatbuf_t *statbuf, const solstat_t *stat);
00118
00119 int readsolstatdata(FILE *fp, gtime_t ts, gtime_t te, double tint,
00120     solstatbuf_t *statbuf);
00121
00122 int readsolstatt(char *files[], int nfile, gtime_t ts, gtime_t te,
00123     double tint, solstatbuf_t *statbuf);
00124
00125 int readsolstat(char *files[], int nfile, solstatbuf_t *statbuf);
00126
00127 int outecef(unsigned char *buff, const char *s, const sol_t *sol,
00128     const solopt_t *opt);
00129
00130 int outpos(unsigned char *buff, const char *s, const sol_t *sol, const solopt_t *opt);
00131
00132 int outenu(unsigned char *buff, const char *s, const sol_t *sol,
00133     const double *rb, const solopt_t *opt);
00134
00135 int outnmea_rmc(unsigned char *buff, const sol_t *sol);
00136
00137 int outnmea_gga(unsigned char *buff, const sol_t *sol);
00138
00139 int outnmea_gsa(unsigned char *buff, const sol_t *sol,
00140     const ssat_t *ssat);
00141
00142 int outnmea_gsv(unsigned char *buff, const sol_t *sol,
00143     const ssat_t *ssat);
00144
00145 int outprcopts(unsigned char *buff, const prcopt_t *opt);
00146
00147 int outsolheads(unsigned char *buff, const solopt_t *opt);
00148
00149 int outsols(unsigned char *buff, const sol_t *sol, const double *rb,
00150     const solopt_t *opt);
00151
00152 int outsolexs(unsigned char *buff, const sol_t *sol, const ssat_t *ssat,
00153     const solopt_t *opt);
00154
00155 void outprcopt(FILE *fp, const prcopt_t *opt);
00156
00157 void outsolhead(FILE *fp, const solopt_t *opt);
00158
00159 void outsol(FILE *fp, const sol_t *sol, const double *rb,
00160     const solopt_t *opt);
00161
00162 void outsolex(FILE *fp, const sol_t *sol, const ssat_t *ssat,
00163     const solopt_t *opt);
00164
00165
00166 /** \} */
00167 /** \} */
00168 #endif  // GNSS_SDR_RTKLIB_SOLUTION_H
```

13.195 rtklib_stream.h File Reference

streaming functions

```
#include "rtklib.h"
#include "rtklib_secure_wipe.h"
```

Macros

- #define TINTACT 200
- #define SERIBUFFSIZE 4096
- #define TIMETAGH_LEN 64
- #define MAXCLI 32
- #define MAXSTATMSG 32
- #define VER_RTKLIB "2.4.2"
- #define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
- #define NTRIP_CLI_PORT 2101
- #define NTRIP_SVR_PORT 80
- #define NTRIP_MAXRSP 32768
- #define NTRIP_MAXSTR 256
- #define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"
- #define NTRIP_RSP_OK_SVR "OK\r\n"
- #define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n"
- #define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
- #define NTRIP_RSP_HTTP "HTTP/"
- #define NTRIP_RSP_ERROR "ERROR"
- #define NTRIP_VERSION_1 1
- #define NTRIP_VERSION_2 2
- #define FTP_CMD "wget"
- #define FTP_TIMEOUT 30

Functions

- serial_t * **openserial** (const char *path, int mode, char *msg)
- void **closeserial** (serial_t *serial)
- int **readserial** (serial_t *serial, unsigned char *buff, int n, char *msg)
- int **writeserial** (serial_t *serial, unsigned char *buff, int n, char *msg)
- int **stateserial** (serial_t *serial)
- int **openfile_** (file_t *file, gtime_t time, char *msg)
- void **closefile_** (file_t *file)
- file_t * **openfile** (const std::string &path, int mode, char *msg)
- void **closefile** (file_t *file)
- void **swapfile** (file_t *file, gtime_t time, char *msg)
- void **swapclose** (file_t *file)
- int **statefile** (file_t *file)
- int **readfile** (file_t *file, unsigned char *buff, int nmax, char *msg)
- int **writefile** (file_t *file, unsigned char *buff, int n, char *msg)
- void **syncfile** (file_t *file1, file_t *file2)
- void **decodetcppath** (const char *path, char *addr, char *port, char *user, char *passwd, char *mntpnt, char *str)
- int **errsock** ()
- int **setsock** (socket_t sock, char *msg)
- socket_t **accept_nb** (socket_t sock, struct sockaddr *addr, socklen_t *len)
- int **connect_nb** (socket_t sock, struct sockaddr *addr, socklen_t len)
- int **recv_nb** (socket_t sock, unsigned char *buff, int n)
- int **send_nb** (socket_t sock, unsigned char *buff, int n)

- int **gentcp** (tcp_t *tcp, int type, char *msg)
- void **discontcp** (tcp_t *tcp, int tcon)
- tcpsvr_t * **opentcpsvr** (const char *path, char *msg)
- void **closetcpsvr** (tcpsvr_t *tcpsvr)
- void **updatetcpsvr** (tcpsvr_t *tcpsvr, char *msg)
- int **accsock** (tcpsvr_t *tcpsvr, char *msg)
- int **waittcpsvr** (tcpsvr_t *tcpsvr, char *msg)
- int **readtcpsvr** (tcpsvr_t *tcpsvr, unsigned char *buff, int n, char *msg)
- int **writetcpsvr** (tcpsvr_t *tcpsvr, unsigned char *buff, int n, char *msg)
- int **statetcpsvr** (tcpsvr_t *tcpsvr)
- int **consock** (tcpcli_t *tcpcli, char *msg)
- tcpcli_t * **opentcpcli** (const char *path, char *msg)
- void **closetcpcli** (tcpcli_t *tcpcli)
- int **waittcpcli** (tcpcli_t *tcpcli, char *msg)
- int **readtcpcli** (tcpcli_t *tcpcli, unsigned char *buff, int n, char *msg)
- int **writetcpcli** (tcpcli_t *tcpcli, unsigned char *buff, int n, char *msg)
- int **statetcpcli** (tcpcli_t *tcpcli)
- int **encbase64** (char *str, const unsigned char *byte, int n)
- int **reqntrip_s** (ntrip_t *ntrip, char *msg)
- int **reqntrip_c** (ntrip_t *ntrip, char *msg)
- int **rspntrip_s** (ntrip_t *ntrip, char *msg)
- int **rspntrip_c** (ntrip_t *ntrip, char *msg)
- int **rspntrip_c_v2** (ntrip_t *ntrip, char *msg)
- int **waitntrip** (ntrip_t *ntrip, char *msg)
- ntrip_t * **openntrip** (const char *path, int type, char *msg)
- int **ntripsethost** (ntrip_t *ntrip, const char *hostname, char *msg)
- void **closentrip** (ntrip_t *ntrip)
- int **readntrip** (ntrip_t *ntrip, unsigned char *buff, int n, char *msg)
- int **writentrip** (ntrip_t *ntrip, unsigned char *buff, int n, char *msg)
- int **statentrip** (ntrip_t *ntrip)
- void **decodeftppath** (const char *path, char *addr, char *file, char *user, char *passwd, int *topts)
- gtime_t **nextdltime** (const int *topts, int stat)
- void * **ftpthread** (void *arg)
- ftp_t * **openftp** (const char *path, int type, char *msg)
- void **closeftp** (ftp_t *ftp)
- int **readftp** (ftp_t *ftp, unsigned char *buff, int n, char *msg)
- int **stateftp** (ftp_t *ftp)
- void **strinitcom** ()
- void **strinit** (stream_t *stream)
- int **stropen** (stream_t *stream, int type, int mode, const char *path)
- void **strclose** (stream_t *stream)
- void **strsync** (stream_t *stream1, stream_t *stream2)
- void **strlock** (stream_t *stream)
- void **strunlock** (stream_t *stream)
- int **strread** (stream_t *stream, unsigned char *buff, int n)
- int **strwrite** (stream_t *stream, unsigned char *buff, int n)
- int **strstat** (stream_t *stream, char *msg)
- void **strsum** (stream_t *stream, int *inb, int *inr, int *outb, int *outr)
- void **strsetopt** (const int *opt)
- void **strsettimeout** (stream_t *stream, int inactive_timeout, int tirecon)
- void **strsetdir** (const char *dir)
- void **strsetproxy** (const char *addr)
- gtime_t **strgettime** (stream_t *stream)
- void **strsendnmea** (stream_t *stream, const double *pos)
- int **gen_hex** (const char *msg, unsigned char *buff)
- void **strsendcmd** (stream_t *str, const char *cmd)

13.195.1 Detailed Description

streaming functions

Authors

- 2007-2013, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.

SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_stream.h.

13.195.2 Macro Definition Documentation

13.195.2.1 FTP_CMD

```
#define FTP_CMD "wget"
```
Definition at line 61 of file rtklib_stream.h.

13.195.2.2 FTP_TIMEOUT

```
#define FTP_TIMEOUT 30
```
Definition at line 62 of file rtklib_stream.h.

13.195.2.3 MAXCLI

```
#define MAXCLI 32
```
Definition at line 43 of file rtklib_stream.h.

13.195.2.4 MAXSTATMSG

```
#define MAXSTATMSG 32
```
Definition at line 44 of file rtklib_stream.h.

13.195.2.5 NTRIP_AGENT

```
#define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
```
Definition at line 47 of file rtklib_stream.h.

13.195.2.6 NTRIP_CLI_PORT

```
#define NTRIP_CLI_PORT 2101
```
Definition at line 48 of file rtklib_stream.h.

13.195.2.7 NTRIP_MAXRSP

```
#define NTRIP_MAXRSP 32768
```
Definition at line 50 of file rtklib_stream.h.

13.195.2.8 NTRIP_MAXSTR

```
#define NTRIP_MAXSTR 256
```
Definition at line 51 of file rtklib_stream.h.

13.195.2.9 NTRIP_RSP_ERROR

```
#define NTRIP_RSP_ERROR "ERROR"
```
Definition at line 57 of file rtklib_stream.h.

13.195.2.10 NTRIP_RSP_HTTP

```
#define NTRIP_RSP_HTTP "HTTP/"
```
Definition at line 56 of file rtklib_stream.h.

13.195.2.11 NTRIP_RSP_OK_CLI

```
#define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"
```
Definition at line 52 of file rtklib_stream.h.

13.195.2.12 NTRIP_RSP_OK_SVR

```
#define NTRIP_RSP_OK_SVR "OK\r\n"
```
Definition at line 53 of file rtklib_stream.h.

13.195.2.13 NTRIP_RSP_SRCTBL

```
#define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n"
```
Definition at line 54 of file rtklib_stream.h.

13.195.2.14 NTRIP_RSP_TBLEND

```
#define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
```
Definition at line 55 of file rtklib_stream.h.

13.195.2.15 NTRIP_SVR_PORT

```
#define NTRIP_SVR_PORT 80
```
Definition at line 49 of file rtklib_stream.h.

13.195.2.16 NTRIP_VERSION_1

```
#define NTRIP_VERSION_1 1
```
Definition at line 58 of file rtklib_stream.h.

13.195.2.17 NTRIP_VERSION_2

```
#define NTRIP_VERSION_2 2
```
Definition at line 59 of file rtklib_stream.h.

13.195.2.18 SERIBUFFSIZE

```
#define SERIBUFFSIZE 4096
```
Definition at line 41 of file rtklib_stream.h.

13.195.2.19 TIMETAGH_LEN

```
#define TIMETAGH_LEN 64
```
Definition at line 42 of file rtklib_stream.h.

13.195.2.20 TINTACT

```
#define TINTACT 200
```
Definition at line 40 of file rtklib_stream.h.

13.195.2.21 VER_RTKLIB

```
#define VER_RTKLIB "2.4.2"
```

Definition at line 46 of file rtklib_stream.h.

13.196 rtklib_stream.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_stream.h
00003  * \brief streaming functions
00004  * \authors <ul>
00005  *          <li> 2007-2013, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ----------------------------------------------------------------------------
00023  * Copyright (C) 2007-2013, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  *----------------------------------------------------------------------------*/
00031
00032 #ifndef GNSS_SDR_RTKLIB_STREAM_H
00033 #define GNSS_SDR_RTKLIB_STREAM_H
00034
00035 #include "rtklib.h"
00036 #include "rtklib_secure_wipe.h"
00037
00038 /* constants -----------------------------------------------------------------*/
00039
00040 #define TINTACT 200       /* period for stream active (ms) */
00041 #define SERIBUFFSIZE 4096 /* serial buffer size (bytes) */
00042 #define TIMETAGH_LEN 64   /* time tag file header length */
00043 #define MAXCLI 32         /* max client connection for tcp svr */
00044 #define MAXSTATMSG 32     /* max length of status message */
00045
00046 #define VER_RTKLIB "2.4.2"
00047 #define NTRIP_AGENT "RTKLIB/" VER_RTKLIB
00048 #define NTRIP_CLI_PORT 2101                     /* default ntrip-client connection port */
00049 #define NTRIP_SVR_PORT 80                       /* default ntrip-server connection port */
00050 #define NTRIP_MAXRSP 32768                      /* max size of ntrip response */
00051 #define NTRIP_MAXSTR 256                        /* max length of mountpoint string */
00052 #define NTRIP_RSP_OK_CLI "ICY 200 OK\r\n"       /* ntrip response: client */
00053 #define NTRIP_RSP_OK_SVR "OK\r\n"               /* ntrip response: server */
00054 #define NTRIP_RSP_SRCTBL "SOURCETABLE 200 OK\r\n" /* ntrip response: source table */
00055 #define NTRIP_RSP_TBLEND "ENDSOURCETABLE"
00056 #define NTRIP_RSP_HTTP "HTTP/"  /* ntrip response: http */
00057 #define NTRIP_RSP_ERROR "ERROR" /* ntrip response: error */
00058 #define NTRIP_VERSION_1 1
00059 #define NTRIP_VERSION_2 2
00060
00061 #define FTP_CMD "wget" /* ftp/http command */
00062 #define FTP_TIMEOUT 30 /* ftp/http timeout (s) */
00063
00064
00065 serial_t *openserial(const char *path, int mode, char *msg);
00066
00067 void closeserial(serial_t *serial);
00068
00069 int readserial(serial_t *serial, unsigned char *buff, int n, char *msg);
00070
00071 int writeserial(serial_t *serial, unsigned char *buff, int n, char *msg);
00072
00073 int stateserial(serial_t *serial);
00074
00075 int openfile_(file_t *file, gtime_t time, char *msg);
```

```
00076
00077 void closefile_(file_t *file);
00078
00079 file_t *openfile(const std::string &path, int mode, char *msg);
00080
00081 void closefile(file_t *file);
00082
00083 void swapfile(file_t *file, gtime_t time, char *msg);
00084
00085 void swapclose(file_t *file);
00086
00087 int statefile(file_t *file);
00088
00089 int readfile(file_t *file, unsigned char *buff, int nmax, char *msg);
00090
00091 int writefile(file_t *file, unsigned char *buff, int n, char *msg);
00092
00093 void syncfile(file_t *file1, file_t *file2);
00094
00095 void decodetcppath(const char *path, char *addr, char *port, char *user,
00096     char *passwd, char *mntpnt, char *str);
00097
00098 int errsock();
00099
00100 int setsock(socket_t sock, char *msg);
00101
00102 socket_t accept_nb(socket_t sock, struct sockaddr *addr, socklen_t *len);
00103
00104 int connect_nb(socket_t sock, struct sockaddr *addr, socklen_t len);
00105
00106 int recv_nb(socket_t sock, unsigned char *buff, int n);
00107
00108 int send_nb(socket_t sock, unsigned char *buff, int n);
00109
00110 int gentcp(tcp_t *tcp, int type, char *msg);
00111
00112 void discontcp(tcp_t *tcp, int tcon);
00113
00114 tcpsvr_t *opentcpsvr(const char *path, char *msg);
00115
00116 void closetcpsvr(tcpsvr_t *tcpsvr);
00117
00118 void updatetcpsvr(tcpsvr_t *tcpsvr, char *msg);
00119
00120 int accsock(tcpsvr_t *tcpsvr, char *msg);
00121
00122 int waittcpsvr(tcpsvr_t *tcpsvr, char *msg);
00123
00124 int readtcpsvr(tcpsvr_t *tcpsvr, unsigned char *buff, int n, char *msg);
00125
00126 int writetcpsvr(tcpsvr_t *tcpsvr, unsigned char *buff, int n, char *msg);
00127
00128 int statetcpsvr(tcpsvr_t *tcpsvr);
00129
00130 int consock(tcpcli_t *tcpcli, char *msg);
00131
00132 tcpcli_t *opentcpcli(const char *path, char *msg);
00133
00134 void closetcpcli(tcpcli_t *tcpcli);
00135
00136 int waittcpcli(tcpcli_t *tcpcli, char *msg);
00137
00138 int readtcpcli(tcpcli_t *tcpcli, unsigned char *buff, int n, char *msg);
00139
00140 int writetcpcli(tcpcli_t *tcpcli, unsigned char *buff, int n, char *msg);
00141
00142 int statetcpcli(tcpcli_t *tcpcli);
00143
00144 int encbase64(char *str, const unsigned char *byte, int n);
00145
00146 int reqntrip_s(ntrip_t *ntrip, char *msg);
00147
00148 int reqntrip_c(ntrip_t *ntrip, char *msg);
00149
00150 int rspntrip_s(ntrip_t *ntrip, char *msg);
00151
00152 int rspntrip_c(ntrip_t *ntrip, char *msg);
00153
00154 int rspntrip_c_v2(ntrip_t *ntrip, char *msg);
00155
00156 int waitntrip(ntrip_t *ntrip, char *msg);
00157
00158 ntrip_t *opennntrip(const char *path, int type, char *msg);
00159
00160 int ntripsethost(ntrip_t *ntrip, const char *hostname, char *msg);
00161
00162 void closentrip(ntrip_t *ntrip);
```

```
00163
00164 int readntrip(ntrip_t *ntrip, unsigned char *buff, int n, char *msg);
00165
00166 int writentrip(ntrip_t *ntrip, unsigned char *buff, int n, char *msg);
00167
00168 int statentrip(ntrip_t *ntrip);
00169
00170 void decodeftppath(const char *path, char *addr, char *file, char *user,
00171     char *passwd, int *topts);
00172
00173 gtime_t nextdltime(const int *topts, int stat);
00174
00175 void *ftpthread(void *arg);
00176
00177 ftp_t *openftp(const char *path, int type, char *msg);
00178
00179 void closeftp(ftp_t *ftp);
00180
00181 int readftp(ftp_t *ftp, unsigned char *buff, int n, char *msg);
00182
00183 int stateftp(ftp_t *ftp);
00184
00185 void strinitcom();
00186
00187 void strinit(stream_t *stream);
00188
00189 int stropen(stream_t *stream, int type, int mode, const char *path);
00190
00191 void strclose(stream_t *stream);
00192
00193 void strsync(stream_t *stream1, stream_t *stream2);
00194
00195 void strlock(stream_t *stream);
00196
00197 void strunlock(stream_t *stream);
00198
00199 int strread(stream_t *stream, unsigned char *buff, int n);
00200
00201 int strwrite(stream_t *stream, unsigned char *buff, int n);
00202
00203 int strstat(stream_t *stream, char *msg);
00204
00205 void strsum(stream_t *stream, int *inb, int *inr, int *outb, int *outr);
00206
00207 void strsetopt(const int *opt);
00208
00209 void strsettimeout(stream_t *stream, int inactive_timeout, int tirecon);
00210
00211 void strsetdir(const char *dir);
00212
00213 void strsetproxy(const char *addr);
00214
00215 gtime_t strgettime(stream_t *stream);
00216
00217 void strsendnmea(stream_t *stream, const double *pos);
00218
00219 int gen_hex(const char *msg, unsigned char *buff);
00220
00221 void strsendcmd(stream_t *str, const char *cmd);
00222
00223
00224 #endif
```

13.197 rtklib_tides.h File Reference

Tidal displacement corrections.

`#include "rtklib.h"`

Functions

- void **tide_pl** (const double *eu, const double *rp, double GMp, const double *pos, double *dr)
- void **tide_solid** (const double *rsun, const double *rmoon, const double *pos, const double *E, double gmst, int opt, double *dr)
- void **tide_oload** (gtime_t tut, const double *odisp, double *denu)
- void **iers_mean_pole** (gtime_t tut, double *xp_bar, double *yp_bar)
- void **tide_pole** (gtime_t tut, const double *pos, const double *erpv, double *denu)
- void **tidedisp** (gtime_t tutc, const double *rr, int opt, const erp_t *erp, const double *odisp, double *dr)

Variables

- const double GME = 3.986004415E+14
- const double GMS = 1.327124E+20
- const double GMM = 4.902801E+12

13.197.1 Detailed Description

Tidal displacement corrections.

Authors

- 2015, T. Takasu
- 2017, Javier Arribas
- 2017, Carles Fernandez

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.
Copyright (C) 2015, T. Takasu Copyright (C) 2017, Javier Arribas Copyright (C) 2017, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause
References: [1] D.D.McCarthy, IERS Technical Note 21, IERS Conventions 1996, July 1996 [2] D.D.McCarthy and G.Petit, IERS Technical Note 32, IERS Conventions 2003, November 2003 [3] D.A.Vallado, Fundamentals of Astrodynamics and Applications 2nd ed, Space Technology Library, 2004 [4] J.Kouba, A Guide to using International GNSS Service (IGS) products, May 2009 [5] G.Petit and B.Luzum (eds), IERS Technical Note No. 36, IERS

13.197.1.1 Conventions (2010), 2010

Definition in file rtklib_tides.h.

13.197.2 Variable Documentation

13.197.2.1 GME

```
const double GME = 3.986004415E+14
```
Definition at line 51 of file rtklib_tides.h.

13.197.2.2 GMM

```
const double GMM = 4.902801E+12
```
Definition at line 53 of file rtklib_tides.h.

13.197.2.3 GMS

```
const double GMS = 1.327124E+20
```
Definition at line 52 of file rtklib_tides.h.

13.198 rtklib_tides.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_tides.h
00003  * \brief Tidal displacement corrections
00004  * \authors <ul>
00005  *          <li> 2015, T. Takasu
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017, Carles Fernandez
00008  *          </ul>
```

```
00009  *
00010  * This is a derived work from RTKLIB http://www.rtklib.com/
00011  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00012  * released under the BSD 2-clause license with an additional exclusive clause
00013  * that does not apply here. This additional clause is reproduced below:
00014  *
00015  * " The software package includes some companion executive binaries or shared
00016  * libraries necessary to execute APs on Windows. These licenses succeed to the
00017  * original ones of these software. "
00018  *
00019  * Neither the executive binaries nor the shared libraries are required by, used
00020  * or included in GNSS-SDR.
00021  *
00022  * ------------------------------------------------------------------------------
00023  * Copyright (C) 2015, T. Takasu
00024  * Copyright (C) 2017, Javier Arribas
00025  * Copyright (C) 2017, Carles Fernandez
00026  * All rights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-2-Clause
00029  *
00030  * References:
00031  *     [1] D.D.McCarthy, IERS Technical Note 21, IERS Conventions 1996, July 1996
00032  *     [2] D.D.McCarthy and G.Petit, IERS Technical Note 32, IERS Conventions
00033  *         2003, November 2003
00034  *     [3] D.A.Vallado, Fundamentals of Astrodynamics and Applications 2nd ed,
00035  *         Space Technology Library, 2004
00036  *     [4] J.Kouba, A Guide to using International GNSS Service (IGS) products,
00037  *         May 2009
00038  *     [5] G.Petit and B.Luzum (eds), IERS Technical Note No. 36, IERS
00039  *         Conventions (2010), 2010
00040  * -----------------------------------------------------------------------------
00041  */
00042
00043
00044 #ifndef GNSS_SDR_RTKLIB_TIDES_H
00045 #define GNSS_SDR_RTKLIB_TIDES_H
00046
00047
00048 #include "rtklib.h"
00049
00050
00051 const double GME = 3.986004415E+14; /* earth gravitational constant */
00052 const double GMS = 1.327124E+20;    /* sun gravitational constant */
00053 const double GMM = 4.902801E+12;    /* moon gravitational constant */
00054
00055 void tide_pl(const double *eu, const double *rp, double GMp,
00056     const double *pos, double *dr);
00057
00058 void tide_solid(const double *rsun, const double *rmoon,
00059     const double *pos, const double *E, double gmst, int opt,
00060     double *dr);
00061
00062 void tide_oload(gtime_t tut, const double *odisp, double *denu);
00063
00064 void iers_mean_pole(gtime_t tut, double *xp_bar, double *yp_bar);
00065
00066
00067 void tide_pole(gtime_t tut, const double *pos, const double *erpv,
00068     double *denu);
00069
00070 void tidedisp(gtime_t tutc, const double *rr, int opt, const erp_t *erp,
00071     const double *odisp, double *dr);
00072 #endif
```

13.199 rtklib_tls.h File Reference

TLS 1.2-or-newer client transport for RTKLIB streams.

```
#include "rtklib.h"
#include <memory>
#include <string>
```

Classes

- class Rtklib_Tls_Client

13.199.1 Detailed Description

TLS 1.2-or-newer client transport for RTKLIB streams.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2026 Carles Fernandez-Prades SPDX-License-Identifier: BSD-2-Clause
Definition in file rtklib_tls.h.

13.200 rtklib_tls.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_tls.h
00003  * \brief TLS 1.2-or-newer client transport for RTKLIB streams
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2026 Carles Fernandez-Prades
00012  * SPDX-License-Identifier: BSD-2-Clause
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_RTKLIB_TLS_H
00018 #define GNSS_SDR_RTKLIB_TLS_H
00019
00020 #include "rtklib.h"
00021 #include <memory>
00022 #include <string>
00023
00024 class Rtklib_Tls_Client_Test_Access;
00025
00026 class Rtklib_Tls_Client
00027 {
00028 public:
00029     explicit Rtklib_Tls_Client(std::string hostname);
00030     ~Rtklib_Tls_Client();
00031
00032     Rtklib_Tls_Client(const Rtklib_Tls_Client &) = delete;
00033     Rtklib_Tls_Client &operator=(const Rtklib_Tls_Client &) = delete;
00034     Rtklib_Tls_Client(Rtklib_Tls_Client &&) = delete;
00035     Rtklib_Tls_Client &operator=(Rtklib_Tls_Client &&) = delete;
00036
00037     bool initialize(char *msg);
00038     void reset();
00039
00040     // Replace the host name used for SNI and certificate verification. The
00041     // SSL context and CA store are host-independent, so an initialized client
00042     // can be retargeted without rebuilding them. Fails (returns false) once a
00043     // TLS session exists: the hostname is consumed when the handshake starts.
00044     bool set_hostname(const std::string &hostname);
00045
00046     // Return 1 when the TLS session is established, 0 while a non-blocking
00047     // handshake is in progress, and -1 on a fatal error.
00048     int handshake(socket_t sock, char *msg);
00049
00050     // Return bytes transferred, 0 when the operation would block, and -1 on a
00051     // fatal TLS error or orderly peer shutdown.
00052     int read(unsigned char *buff, int n, char *msg);
00053     int write(const unsigned char *buff, int n, char *msg);
00054
00055 private:
00056     friend class Rtklib_Tls_Client_Test_Access;
00057
00058     // Test-only trust injection. Production callers always use initialize(),
00059     // which loads the platform trust store.
00060     bool initialize_with_trust_anchor(const std::string &trust_anchor_pem, char *msg);
00061
00062     class Impl;
00063     std::unique_ptr<Impl> d_impl;
00064 };
00065
00066 #endif  // GNSS_SDR_RTKLIB_TLS_H
```

13.201 sbas_signal_replica.h File Reference

This file implements signal generators for SBAS L1 signals.

```
#include <complex>
#include <cstdint>
#include <gsl-lite/gsl-lite.hpp>
```

Functions

- void sbas_l1_code_gen_complex_sampled (own::span< std::complex< float > > dest, uint32_t prn, int32_t sampling_freq)

  Generates complex SBAS L1 C/A code for the desired SV PRN, sampled at sampling_freq [Hz].
- void sbas_l1_code_gen_float (own::span< float > dest, uint32_t prn)

  Generates real SBAS L1 C/A code for the desired SV PRN (chip_shift=0).
- void sbas_l1_code_gen_complex (own::span< std::complex< float > > dest, uint32_t prn, uint32_t chip_shift)

  Generates complex SBAS L1 C/A code for the desired SV PRN with a given chip_shift.

13.201.1 Detailed Description

This file implements signal generators for SBAS L1 signals.

Author

Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es

Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com

SBAS PRN codes (120–138) share the same Gold-code generator as GPS L1 C/A but use SBAS-specific G2 initial state delays. The underlying code generator already handles SBAS PRNs; these wrappers provide a clean SBAS-specific API.

Reference: RTCA DO-229E, Appendix A

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sbas_signal_replica.h.

13.202 sbas_signal_replica.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sbas_signal_replica.h
00003  * \brief This file implements signal generators for SBAS L1 signals
00004  * \author Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es
00005  * \author Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com
00006  *
00007  * SBAS PRN codes (120-138) share the same Gold-code generator as GPS L1 C/A
00008  * but use SBAS-specific G2 initial state delays. The underlying code generator
00009  * already handles SBAS PRNs; these wrappers provide a clean SBAS-specific API.
00010  *
00011  * Reference: RTCA DO-229E, Appendix A
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024
00025 #ifndef GNSS_SDR_SBAS_SIGNAL_REPLICA_H
00026 #define GNSS_SDR_SBAS_SIGNAL_REPLICA_H
00027
00028 #include <complex>
00029 #include <cstdint>
```

```
00030 #if HAS_STD_SPAN
00031 #include <span>
00032 namespace own = std;
00033 #else
00034 #include <gsl-lite/gsl-lite.hpp>
00035 namespace own = gsl_lite;
00036 #endif
00037
00038 /** \addtogroup Algorithms_Library
00039  * \{ */
00040 /** \addtogroup Algorithm_libs algorithms_libs
00041  * \{ */
00042
00043 //! Generates complex SBAS L1 C/A code for the desired SV PRN, sampled at sampling_freq [Hz]
00044 void sbas_l1_code_gen_complex_sampled(own::span<std::complex<float>> dest, uint32_t prn, int32_t
      sampling_freq);
00045
00046 //! Generates real SBAS L1 C/A code for the desired SV PRN (chip_shift=0)
00047 void sbas_l1_code_gen_float(own::span<float> dest, uint32_t prn);
00048
00049 //! Generates complex SBAS L1 C/A code for the desired SV PRN with a given chip_shift
00050 void sbas_l1_code_gen_complex(own::span<std::complex<float>> dest, uint32_t prn, uint32_t chip_shift);
00051
00052 /** \} */
00053 /** \} */
00054
00055 #endif  // GNSS_SDR_SBAS_SIGNAL_REPLICA_H
```

13.203 sensor_data_aggregator.h File Reference

Aggregates sensor samples from gnu radio stream tags into typed lists for easy access.

```
#include "sensor_data_source_configuration.h"
#include "sensor_identifier.h"
#include <gnuradio/tags.h>
#include <cstdint>
#include <gnss_block_interface.h>
#include <vector>
```

Classes

- struct SensorDataSample< DataType >
- class SensorDataAggregator

13.203.1 Detailed Description

Aggregates sensor samples from gnu radio stream tags into typed lists for easy access.

Author

Victor Castillo, 2024. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sensor_data_aggregator.h.

13.204 sensor_data_aggregator.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_aggregator.h
00003  * \brief  Aggregates sensor samples from gnu radio stream tags into typed lists for easy access
00004  * \author Victor Castillo, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
```

```
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_AGGREGATOR_H
00019 #define GNSS_SDR_SENSOR_DATA_AGGREGATOR_H
00020
00021 #include "sensor_data_source_configuration.h"
00022 #include "sensor_identifier.h"
00023 #include <gnuradio/tags.h>
00024 #include <cstdint>
00025 #include <gnss_block_interface.h>
00026 #include <vector>
00027
00028 /** \addtogroup Algorithms_Library
00029  * \{ */
00030 /** \addtogroup Algorithm_libs algorithms_libs
00031  * \{ */
00032
00033
00034 template <typename DataType>
00035 struct SensorDataSample
00036 {
00037     uint64_t timestamp;
00038     DataType value;
00039     constexpr SensorDataSample() = default;
00040     constexpr SensorDataSample(uint64_t t, DataType v)
00041         : timestamp(t), value(v) {}
00042 };
00043
00044
00045 class SensorDataAggregator
00046 {
00047 public:
00048     const pmt::pmt_t SENSOR_DATA_TAG = pmt::mp("sensor_data");
00049
00050     explicit SensorDataAggregator(const SensorDataSourceConfiguration& configuration, const
      std::vector<SensorIdentifier::value_type>& required_sensors);
00051
00052     void update(const std::vector<gr::tag_t>& tags);
00053
00054     const std::vector<SensorDataSample<float»& get_f32(SensorIdentifier::value_type sensor_id) const;
00055     SensorDataSample<float> get_last_f32(SensorIdentifier::value_type sensor_id) const;
00056     SensorDataSample<float> get_average_f32(SensorIdentifier::value_type sensor_id) const;
00057
00058     const std::vector<SensorDataSample<double»& get_f64(SensorIdentifier::value_type sensor_id) const;
00059     SensorDataSample<double> get_last_f64(SensorIdentifier::value_type sensor_id) const;
00060     SensorDataSample<double> get_average_f64(SensorIdentifier::value_type sensor_id) const;
00061
00062     // More getters to be added in the future for different types
00063 private:
00064     void append_data(const pmt::pmt_t& data_dict);
00065
00066     std::unordered_map<SensorIdentifier::value_type, std::vector<SensorDataSample<float»> f32_data_{};
00067     std::unordered_map<SensorIdentifier::value_type, std::vector<SensorDataSample<double»>
      f64_data_{};
00068     // More maps to be added in the future for different types
00069 };
00070
00071
00072 /** \} */
00073 /** \} */
00074 #endif  // GNSS_SDR_SENSOR_DATA_AGGREGATOR_H
```

13.205 sensor_data_file.h File Reference

Provides a simple abstraction for reading contiguous binary data from a file.

```
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <gnss_block_interface.h>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class SensorDataFile

13.205.1 Detailed Description

Provides a simple abstraction for reading contiguous binary data from a file.

Author

Victor Castillo, 2024. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file sensor_data_file.h.

13.206 sensor_data_file.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_file.h
00003  * \brief  Provides a simple abstraction for reading contiguous binary data from a file
00004  * \author Victor Castillo, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_FILE_H
00019 #define GNSS_SDR_SENSOR_DATA_FILE_H
00020
00021 #include <cstddef>  // for size_t
00022 #include <cstdint>
00023 #include <fstream>
00024 #include <gnss_block_interface.h>
00025 #include <memory>
00026 #include <string>
00027 #include <vector>
00028
00029 /** \addtogroup Algorithms_Library
00030  * \{ */
00031 /** \addtogroup Algorithm_libs algorithms_libs
00032  * \{ */
00033
00034
00035 class SensorDataFile
00036 {
00037 public:
00038     using sptr = gnss_shared_ptr<SensorDataFile>;
00039     using id_type = std::size_t;
00040
00041     SensorDataFile(
00042         const std::string& path,
00043         const std::size_t& sample_delay,
00044         const std::size_t& sample_period,
00045         const std::size_t& offset_in_file,
00046         const std::size_t& item_size,
00047         const bool& repeat);
00048
00049     void reset();
00050
00051     bool read_until_sample(std::size_t end_sample, std::size_t& sample_stamp, std::vector<uint8_t>&
      buffer);
00052
00053     std::size_t get_chunks_read() const;
00054
00055 private:
00056     bool read_item(std::vector<uint8_t>& buffer);
00057
00058     void read_into_io_buffer();
00059
00060     void read_into_item_buffer(std::vector<uint8_t>& item_buf);
```

```
00061
00062     std::string path_;
00063     std::ifstream file_;
00064     std::size_t sample_period_;
00065     std::size_t offset_in_file_;
00066     std::size_t item_size_;
00067
00068     std::size_t chunks_read_;
00069     std::size_t next_sample_stamp_;
00070     std::vector<uint8_t> io_buffer_;
00071     std::size_t io_buffer_size_;
00072     std::size_t offset_in_io_buffer_;
00073     bool repeat_;
00074     bool done_;
00075 };
00076
00077 /** \} */
00078 /** \} */
00079 #endif  // GNSS_SDR_SENSOR_DATA_FILE_H
```

13.207 sensor_data_resampler.h File Reference

Updates timestamp within sensor data tags. To be used in resampler blocks.

```
#include <gnuradio/tags.h>
#include <vector>
```

Functions

- std::vector< gr::tag_t > resample_sensor_data_tags (const std::vector< gr::tag_t > &tags, double freq_in, double freq_out)

  Updates timestamp within sensor data tags. To be used in resampler blocks.

13.207.1 Detailed Description

Updates timestamp within sensor data tags. To be used in resampler blocks.

Author

Victor Castillo, 2025. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sensor_data_resampler.h.

13.208 sensor_data_resampler.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_resampler.h
00003  * \brief  Updates timestamp within sensor data tags. To be used in resampler blocks.
00004  * \author Victor Castillo, 2025. victorcastilloaguero(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_RESAMPLER_H
00019 #define GNSS_SDR_SENSOR_DATA_RESAMPLER_H
00020
00021 #include <gnuradio/tags.h>
00022 #include <vector>
00023
00024 /** \addtogroup Algorithms_Library
```

```
00025  * \{ */
00026 /** \addtogroup Algorithm_libs algorithms_libs
00027  * \{ */
00028
00029
00030 /** \brief Updates timestamp within sensor data tags. To be used in resampler blocks.
00031  *
00032  *  \param tags Stream tags as retrieved by `get_tags_in_window' or `get_tags_in_range'.
00033  *  \param freq_in Input RF sample rate.
00034  *  \param freq_out Output RF sample rate.
00035  *  \return Stream tags to be added with `add_item_tag'.
00036  */
00037 std::vector<gr::tag_t> resample_sensor_data_tags(const std::vector<gr::tag_t>& tags, double freq_in,
      double freq_out);
00038
00039
00040 /** \} */
00041 /** \} */
00042 #endif  // GNSS_SDR_SENSOR_DATA_RESAMPLER_H
```

13.209 sensor_data_source.h File Reference

GNURadio block that adds extra data to the sample stream.

```
#include "gnss_block_interface.h"
#include "sensor_data/sensor_data_file.h"
#include "sensor_data/sensor_data_source_configuration.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
#include <cstddef>
```

Classes

- class SensorDataSource

13.209.1 Detailed Description

GNURadio block that adds extra data to the sample stream.

Author

Victor Castillo, 2024. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file sensor_data_source.h.

13.210 sensor_data_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_source.h
00003  * \brief  GNURadio block that adds extra data to the sample stream.
00004  * \author Victor Castillo, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_SOURCE_H
00019 #define GNSS_SDR_SENSOR_DATA_SOURCE_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "sensor_data/sensor_data_file.h"
```

```
00023 #include "sensor_data/sensor_data_source_configuration.h"
00024 #include <gnuradio/sync_block.h>  // for sync_block
00025 #include <gnuradio/types.h>       // for gr_vector_const_void_star
00026 #include <cstddef>                // for size_t
00027
00028 /** \addtogroup Algorithms_Library
00029  * \{ */
00030 /** \addtogroup Algorithm_libs algorithms_libs
00031  * \{ */
00032
00033 class SensorDataSource : public gr::sync_block
00034 {
00035 public:
00036     using sptr = gnss_shared_ptr<SensorDataSource>;
00037
00038     SensorDataSource(
00039         const SensorDataSourceConfiguration& configuration,
00040         const gr::io_signature::sptr& io_signature);
00041
00042     int work(int noutput_items,
00043         gr_vector_const_void_star& input_items,
00044         gr_vector_void_star& output_items) override;
00045
00046 private:
00047     std::unordered_map<SensorDataFile::id_type, SensorDataFile::sptr> sensor_data_files_;
00048     std::unordered_map<SensorDataFile::id_type, std::vector<SensorDataConfiguration»
      sensor_config_map_;
00049     std::size_t item_size_;
00050     std::size_t items_per_sample_;
00051 };
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_SENSOR_DATA_SOURCE_H
```

13.211 sensor_data_source_configuration.h File Reference

```
#include "configuration_interface.h"
#include "sensor_data_type.h"
#include "sensor_identifier.h"
#include <string>
#include <unordered_map>
```

Classes

- struct SensorDataFileConfiguration
- struct SensorDataConfiguration
- class SensorDataSourceConfiguration

13.211.1 Detailed Description

Author

Victor Castillo, 2024. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sensor_data_source_configuration.h.

13.212 sensor_data_source_configuration.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_source_configuration.h
00003  * \brief
00004  * \author Victor Castillo, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
```

```
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_SOURCE_CONFIGURATION_H
00019 #define GNSS_SDR_SENSOR_DATA_SOURCE_CONFIGURATION_H
00020
00021 #include "configuration_interface.h"
00022 #include "sensor_data_type.h"
00023 #include "sensor_identifier.h"
00024 #include <string>
00025 #include <unordered_map>
00026
00027 /** \addtogroup Algorithms_Library
00028  * \{ */
00029 /** \addtogroup Algorithm_libs algorithms_libs
00030  * \{ */
00031
00032 class ConfigurationInterface;
00033
00034 struct SensorDataFileConfiguration
00035 {
00036     uint64_t id;
00037     uint64_t chunk_size;
00038     uint64_t file_offset;
00039     uint64_t sample_offset;
00040     uint64_t sample_period;
00041     std::string filename;
00042     bool repeat;
00043 };
00044
00045
00046 struct SensorDataConfiguration
00047 {
00048     uint64_t id;
00049     uint64_t file_id;
00050     uint64_t offset;
00051     SensorIdentifier::value_type identifier;
00052     SensorDataType::value_type type;
00053     pmt::pmt_t tag_key;
00054 };
00055
00056
00057 class SensorDataSourceConfiguration
00058 {
00059 public:
00060     explicit SensorDataSourceConfiguration(const ConfigurationInterface* configuration);
00061
00062     bool validate() const;
00063
00064     bool is_enabled() const;
00065
00066     bool is_sensor_provided(SensorIdentifier::value_type sensor_id) const;
00067
00068     const std::unordered_map<uint64_t, SensorDataFileConfiguration>& files() const;
00069
00070     const std::vector<SensorDataConfiguration>& sensors() const;
00071
00072     void set_items_per_sample(uint64_t items_per_sample);
00073
00074     uint64_t get_items_per_sample() const;
00075
00076 private:
00077     void configure_files(const ConfigurationInterface* configuration);
00078
00079     void configure_sensors(const ConfigurationInterface* configuration);
00080
00081     bool validate_files() const;
00082
00083     bool validate_sensors() const;
00084
00085     bool enabled_;
00086     std::unordered_map<uint64_t, SensorDataFileConfiguration> files_;
00087     std::vector<SensorDataConfiguration> sensors_;
00088     uint64_t items_per_sample_;
00089 };
00090
00091 /** \} */
00092 /** \} */
00093 #endif  // GNSS_SDR_SENSOR_DATA_SOURCE_CONFIGURATION_H
```

13.213 sensor_data_type.h File Reference

```
#include <pmt/pmt.h>
#include <string>
```

Classes

- struct SensorDataType

13.213.1 Detailed Description

Author

Victor Castillo, 2025. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file sensor_data_type.h.

13.214 sensor_data_type.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_data_type.h
00003  * \brief
00004  * \author Victor Castillo, 2025. victorcastilloaguero(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_DATA_TYPE_H
00019 #define GNSS_SDR_SENSOR_DATA_TYPE_H
00020
00021 #include <pmt/pmt.h>
00022 #include <string>
00023
00024 /** \addtogroup Algorithms_Library
00025  * \{ */
00026 /** \addtogroup Algorithm_libs algorithms_libs
00027  * \{ */
00028
00029 struct SensorDataType
00030 {
00031     SensorDataType() = delete;
00032     enum value_type
00033     {
00034         UINT64,  // Used internally for SAMPLE_STAMP and CHUNK_COUNT
00035         F32,
00036         F64,
00037         I32,
00038         I64,
00039         // More types can be added here, don't forget to update the static functions
00040     };
00041
00042     static value_type from_string(const std::string& s);
00043
00044     static std::string to_string(const value_type& v);
00045
00046     static uint64_t get_size(const value_type& v);
00047
00048     static pmt::pmt_t make_value(const value_type& v, void* value);
00049 };
00050
00051
00052 /** \} */
00053 /** \} */
00054 #endif  // GNSS_SDR_SENSOR_DATA_TYPE_H
```

13.215 sensor_identifier.h File Reference

```
#include "sensor_data_type.h"
#include <functional>
#include <string>
```

Classes

- struct SensorIdentifier
- struct std::hash< SensorIdentifier::value_type >

Namespaces

- namespace std

  STL namespace.

13.215.1 Detailed Description

Author

Victor Castillo, 2025. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2024-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file sensor_identifier.h.

13.216 sensor_identifier.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sensor_identifier.h
00003  * \brief
00004  * \author Victor Castillo, 2025. victorcastilloaguero(at)gmail.com
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2024-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SENSOR_IDENTIFIER_H
00019 #define GNSS_SDR_SENSOR_IDENTIFIER_H
00020
00021 #include "sensor_data_type.h"
00022 #include <functional>
00023 #include <string>
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030 struct SensorIdentifier
00031 {
00032     SensorIdentifier() = delete;
00033     enum value_type : unsigned short
00034     {
00035         SAMPLE_STAMP = 0,  // Used internally
00036         CHUNK_COUNT,       // Used internally
00037         IMU_VEL_X,
00038         IMU_VEL_Y,
00039         IMU_VEL_Z,
00040         IMU_ACC_X,
00041         IMU_ACC_Y,
00042         IMU_ACC_Z,
00043         IMU_ANG_VEL_X,
```

```
00044         IMU_ANG_VEL_Y,
00045         IMU_ANG_VEL_Z,
00046         IMU_ANG_ACC_X,
00047         IMU_ANG_ACC_Y,
00048         IMU_ANG_ACC_Z,
00049     };
00050
00051     static value_type from_string(const std::string& s);
00052
00053     static std::string to_string(value_type v);
00054
00055     static bool is_valid_type(value_type sensor_id, SensorDataType::value_type type);
00056
00057     static SensorDataType::value_type get_internal_type(value_type sensor_id);
00058
00059     static pmt::pmt_t convert_to_internal_type(value_type sensor_id, SensorDataType::value_type
      original_type, const pmt::pmt_t& value);
00060 };
00061
00062
00063 // Fix for C++11
00064 namespace std
00065 {
00066 template <>
00067 struct hash<SensorIdentifier::value_type>
00068 {
00069     std::size_t operator()(const SensorIdentifier::value_type& key) const noexcept
00070     {
00071         return std::hash<unsigned short>()(static_cast<unsigned short>(key));
00072     }
00073 };
00074 }  // namespace std
00075
00076
00077 /** \} */
00078 /** \} */
00079 #endif  // GNSS_SDR_SENSOR_IDENTIFIER_H
```

13.217 short_x2_to_cshort.h File Reference

Adapts two short streams into a std::complex<short> stream.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
```

Classes

- class short_x2_to_cshort

  This class adapts two short streams into a std::complex<short> stream.

Typedefs

- using short_x2_to_cshort_sptr = gnss_shared_ptr<short_x2_to_cshort>

Functions

- short_x2_to_cshort_sptr **make_short_x2_to_cshort** ()

13.217.1 Detailed Description

Adapts two short streams into a std::complex<short> stream.

Author

Carles Fernandez Prades, cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file short_x2_to_cshort.h.

13.218 short_x2_to_cshort.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file short_x2_to_cshort.h
00003  * \brief Adapts two short streams into a std::complex<short> stream
00004  * \author Carles Fernandez Prades, cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SHORT_X2_TO_CSHORT_H
00018 #define GNSS_SDR_SHORT_X2_TO_CSHORT_H
00019
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024
00025 /** \addtogroup Algorithms_Library
00026  * \{ */
00027 /** \addtogroup Algorithm_libs algorithms_libs
00028  * \{ */
00029
00030
00031 class short_x2_to_cshort;
00032
00033 using short_x2_to_cshort_sptr = gnss_shared_ptr<short_x2_to_cshort>;
00034
00035 short_x2_to_cshort_sptr make_short_x2_to_cshort();
00036
00037 /*!
00038  * \brief This class adapts two short streams into a std::complex<short> stream
00039  */
00040 class short_x2_to_cshort : public gr::sync_block
00041 {
00042 public:
00043     int work(int noutput_items,
00044         gr_vector_const_void_star &input_items,
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     friend short_x2_to_cshort_sptr make_short_x2_to_cshort();
00049     short_x2_to_cshort();
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_SHORT_X2_TO_CSHORT_H
```

13.219 trackingcmd.h File Reference

Class that stores information to update the GNSS signal tracking estimations.

`#include <cstdint>`

Classes

- class TrackingCmd

13.219.1 Detailed Description

Class that stores information to update the GNSS signal tracking estimations.

Author

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file trackingcmd.h.

13.220 trackingcmd.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file trackingcmd.h
00003  * \brief Class that stores information to update the GNSS signal tracking estimations
00004  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_TRACKINGCMD_H_
00019 #define GNSS_SDR_TRACKINGCMD_H_
00020
00021 #include <cstdint>
00022
00023 /** \addtogroup Algorithms_Library
00024  * \{ */
00025 /** \addtogroup Algorithm_libs algorithms_libs
00026  * \{ */
00027
00028 class TrackingCmd
00029 {
00030 public:
00031     TrackingCmd();
00032
00033     bool enable_carrier_nco_cmd = false;
00034     bool enable_code_nco_cmd = false;
00035     double code_freq_chips = 0.0;
00036     double carrier_freq_hz = 0.0;
00037     double carrier_freq_rate_hz_s = 0.0;
00038     uint64_t sample_counter = 0UL;
00039 };
00040
00041 /** \} */
00042 /** \} */
00043 #endif  // GNSS_SDR_TRACKINGCMD_H_
```

13.221 hybrid_observables.h File Reference

Implementation of an adapter of an observables block accepting all kind of signals to a ObservablesInterface.

```
#include "gnss_synchro.h"
#include "hybrid_observables_gs.h"
#include "observables_interface.h"
#include <gnuradio/gr_complex.h>
#include <gnuradio/runtime_types.h>
#include <cstddef>
#include <string>
```

Classes

- class HybridObservables

  This class implements an ObservablesInterface for observables of all kind of GNSS signals.

13.221.1 Detailed Description

Implementation of an adapter of an observables block accepting all kind of signals to a ObservablesInterface.

Author

Mara Branzanti 2013. mara.branzanti(at)gmail.com

Javier Arribas 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file hybrid_observables.h.

13.222 hybrid_observables.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file hybrid_observables.h
00003  * \brief Implementation of an adapter of an observables block accepting all kind
00004  * of signals to a ObservablesInterface
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  * \author Javier Arribas 2013. jarribas(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_HYBRID_OBSERVABLES_H
00021 #define GNSS_SDR_HYBRID_OBSERVABLES_H
00022
00023 #include "gnss_synchro.h"
00024 #include "hybrid_observables_gs.h"
00025 #include "observables_interface.h"
00026 #include <gnuradio/gr_complex.h>     // for gr_complex
00027 #include <gnuradio/runtime_types.h>  // for basic_block_sptr, top_block_sptr
00028 #include <cstddef>
00029 #include <string>
00030
00031 /** \addtogroup Observables
00032  * Classes for the computation of GNSS observables
00033  * \{ */
00034 /** \addtogroup Observables_adapters obs_adapters
00035  * Wrap GNU Radio observables blocks with an ObservablesInterface
00036  * \{ */
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class implements an ObservablesInterface for observables of all kind of GNSS signals
00042  */
00043 class HybridObservables : public ObservablesInterface
00044 {
00045 public:
00046     HybridObservables(const ConfigurationInterface* configuration,
00047         const std::string& role,
00048         unsigned int in_streams,
00049         unsigned int out_streams);
00050
00051     ~HybridObservables() = default;
00052
00053     inline std::string role() override
00054     {
00055         return role_;
00056     }
00057
00058     //!  Returns "Hybrid_Observables"
00059     inline std::string implementation() override
00060     {
00061         return "Hybrid_Observables";
00062     }
00063
00064     void connect(gr::top_block_sptr top_block) override;
00065     void disconnect(gr::top_block_sptr top_block) override;
```

```
00066     gr::basic_block_sptr get_left_block() override;
00067     gr::basic_block_sptr get_right_block() override;
00068
00069     inline void reset() override
00070     {
00071         return;
00072     }
00073
00074     //! All blocks must have an item_size() function implementation
00075     inline size_t item_size() override
00076     {
00077         return sizeof(Gnss_Synchro);
00078     }
00079
00080 private:
00081     hybrid_observables_gs_sptr observables_;
00082     std::string dump_filename_;
00083     std::string role_;
00084     unsigned int in_streams_;
00085     unsigned int out_streams_;
00086     bool dump_;
00087     bool dump_mat_;
00088 };
00089
00090 /** \} */
00091 /** \} */
00092 #endif
```

13.223 hybrid_observables_gs.h File Reference

Interface of the observables computation block.

```
#include "gnss_block_interface.h"
#include "gnss_time.h"
#include "obs_conf.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <memory>
#include <queue>
#include <string>
#include <typeinfo>
#include <vector>
```

Classes

- class hybrid_observables_gs

  This class implements a block that computes observables.

Typedefs

- using hybrid_observables_gs_sptr = gnss_shared_ptr<hybrid_observables_gs>

Functions

- hybrid_observables_gs_sptr **hybrid_observables_gs_make** (const Obs_Conf &conf_)

13.223.1 Detailed Description

Interface of the observables computation block.

Author

Mara Branzanti 2013. mara.branzanti(at)gmail.com

Javier Arribas 2013. jarribas(at)cttc.es

Antonio Ramos 2018. antonio.ramos(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file hybrid_observables_gs.h.

13.224 hybrid_observables_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file hybrid_observables_gs.h
00003  * \brief Interface of the observables computation block
00004  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00005  * \author Javier Arribas 2013. jarribas(at)cttc.es
00006  * \author Antonio Ramos 2018. antonio.ramos(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_HYBRID_OBSERVABLES_GS_H
00021 #define GNSS_SDR_HYBRID_OBSERVABLES_GS_H
00022
00023 #include "gnss_block_interface.h"
00024 #include "gnss_time.h"  // for timetags produced by Tracking
00025 #include "obs_conf.h"
00026 #include <boost/circular_buffer.hpp>  // for boost::circular_buffer
00027 #include <gnuradio/block.h>           // for block
00028 #include <gnuradio/types.h>           // for gr_vector_int
00029 #include <cstddef>                    // for size_t
00030 #include <cstdint>                    // for int32_t
00031 #include <fstream>                    // for std::ofstream
00032 #include <memory>                     // for std::shared, std:unique_ptr
00033 #include <queue>                      // for std::queue
00034 #include <string>                     // for std::string
00035 #include <typeinfo>                   // for typeid
00036 #include <vector>                     // for std::vector
00037
00038 /** \addtogroup Observables
00039  * \{ */
00040 /** \addtogroup Observables_gnuradio_blocks obs_gr_blocks
00041  * GNU Radio blocks for the computation of GNSS observables
00042  * \{ */
00043
00044
00045 class Gnss_Satellite;
00046 class Gnss_Synchro;
00047 class hybrid_observables_gs;
00048
00049 template <class T>
00050 class Gnss_circular_deque;
00051
00052 using hybrid_observables_gs_sptr = gnss_shared_ptr<hybrid_observables_gs>;
00053
00054 hybrid_observables_gs_sptr hybrid_observables_gs_make(const Obs_Conf& conf_);
00055
00056 /*!
00057  * \brief This class implements a block that computes observables
00058  */
00059 class hybrid_observables_gs : public gr::block
00060 {
00061 public:
00062     ~hybrid_observables_gs() noexcept;
00063     void forecast(int noutput_items, gr_vector_int& ninput_items_required);
00064     int general_work(int noutput_items, gr_vector_int& ninput_items,
00065         gr_vector_const_void_star& input_items, gr_vector_void_star& output_items);
00066
00067 private:
00068     friend hybrid_observables_gs_sptr hybrid_observables_gs_make(const Obs_Conf& conf_);
```

```
00069
00070     explicit hybrid_observables_gs(const Obs_Conf& conf_);
00071
00072     const size_t d_double_type_hash_code = typeid(double).hash_code();
00073     const size_t d_int_type_hash_code = typeid(int).hash_code();
00074
00075     static Gnss_Satellite pretty_satellite(char system, uint32_t prn);
00076
00077     // Shortest interruption of the observation stream of a channel that can be
00078     // attributed to a loss of lock, in seconds. Reacquiring a satellite and
00079     // decoding its telemetry again takes seconds, so shorter interruptions are
00080     // transient failures to interpolate the tracking history and leave the
00081     // carrier phase untouched.
00082     static constexpr double MIN_REACQUISITION_GAP_S = 1.0;
00083
00084     // Tells whether the carrier phase ambiguity of a channel may have changed
00085     // since its previous valid observation: the accumulated carrier phase
00086     // restarts from scratch whenever the channel (re)acquires a satellite, so
00087     // an observation that resumes after a loss of lock carries a new ambiguity.
00088     // Epochs are counts of observation epochs, epoch_interval_s apart.
00089     static bool phase_stream_is_discontinuous(bool has_previous_observation,
00090         uint64_t last_valid_epoch,
00091         uint64_t current_epoch,
00092         uint32_t last_valid_prn,
00093         uint32_t current_prn,
00094         double epoch_interval_s);
00095
00096     // Tells whether the carrier phase has stepped by half a cycle because the
00097     // Costas loop ambiguity resolved by the Telemetry Decoder changed: it adds
00098     // half a cycle to the carrier phase while the PLL is locked at 180 degrees.
00099     // A new ambiguity (carrier_phase_discontinuous) supersedes this report.
00100     static bool half_cycle_ambiguity_changed(bool has_previous_observation,
00101         bool carrier_phase_discontinuous,
00102         uint32_t last_valid_prn,
00103         uint32_t current_prn,
00104         bool last_pll_180_locked,
00105         bool current_pll_180_locked);
00106
00107     // Linearly interpolates the carrier phase between two tracking samples, in
00108     // the polarity frame of the later one. When the PLL-180 half-cycle
00109     // correction toggles between them, their phases differ by half a cycle on
00110     // top of the true motion, and interpolating across that step would leak a
00111     // fraction of it into the output with no slip flag raised. time_factor is
00112     // the position of the epoch between the two samples, in [0, 1].
00113     static double interpolate_carrier_phase(double phase_early_rads,
00114         bool pll_180_early,
00115         double phase_late_rads,
00116         bool pll_180_late,
00117         double time_factor);
00118
00119     // Longest time between the two tracking samples used to interpolate an
00120     // observation, in seconds. The Telemetry Decoders deliver one sample per
00121     // telemetry symbol, which lasts 20 ms at most (GPS L1 C/A, GPS L2C, Galileo
00122     // E5a, BeiDou D1), so samples further apart are not consecutive: the
00123     // channel stopped delivering valid words in between. For the same reason,
00124     // it is the time by which every channel delivering valid words can be
00125     // interpolated again after the tracking history is emptied.
00126     static constexpr double MAX_INTERPOLATION_INTERVAL_S = 0.05;
00127
00128     // Largest disagreement between the TOW increment and the receiver time
00129     // increment of two consecutive tracking samples, in ms. Both advance at the
00130     // same pace but for the code Doppler and the sampling clock error, which
00131     // amount to microseconds in MAX_INTERPOLATION_INTERVAL_S, while the TOW
00132     // reported by a Telemetry Decoder steps in whole ms.
00133     static constexpr double MAX_TOW_INCREMENT_ERROR_MS = 0.5;
00134
00135     // Largest difference between the TOWs reported at the same epoch by
00136     // channels that agree on the time, in ms. It covers the spread of the
00137     // signal travel times, from MEO satellites at the zenith to geosynchronous
00138     // ones at the horizon (with margin for a receiver in Earth orbit), plus the
00139     // resolution of the TOW, which is reported at the telemetry symbol closest
00140     // to the epoch.
00141     static constexpr int64_t MAX_TOW_SPREAD_MS = 300;
00142
00143     // Largest difference between the receiver time and the TOW on which the
00144     // channels agree, in ms, not counting the rounding of the receiver time to
00145     // the observables interval. It is far above the offsets found in normal
00146     // operation (a signal travel time, plus the drift of a free-running
00147     // receiver clock) and far below the error of a TOW decoded from a false
00148     // frame synchronization.
00149     static constexpr int64_t MAX_RX_TIME_ERROR_MS = 1000;
00150
00151     // Time that the channels need to keep contradicting the receiver time
00152     // before it is set again from them, in seconds.
00153     static constexpr double RX_TIME_CONTRADICTION_TIME_S = 1.0;
00154
00155     // Difference between two TOWs in ms, unwrapped across the end of the week:
```

```
00156     // the result is in [-half a week, half a week).
00157     static int64_t tow_difference_ms(uint32_t tow_ms, uint32_t tow_ref_ms);
00158
00159     // Brings a time into [0, one week) and then rounds it up to a multiple of
00160     // the observables interval, which is how the receiver time is kept. The
00161     // time can be negative or exceed a week, as it happens when a clock offset
00162     // is applied next to the TOW rollover.
00163     static uint32_t align_rx_time_ms(int64_t tow_ms, uint32_t interval_ms);
00164
00165     // Tells whether an observation can be interpolated between two tracking
00166     // samples of a channel: they must be consecutive, and their TOW must have
00167     // advanced as much as the receiver time did. The history of a channel only
00168     // keeps the samples with a valid TOW, so when a channel stops delivering
00169     // them and then resumes with the same satellite (the Telemetry Decoder lost
00170     // and recovered the frame synchronization, or the satellite was
00171     // reacquired), the last sample before the gap and the first one after it
00172     // become neighbors. They do not bracket a continuous piece of signal: their
00173     // carrier phases are unrelated if the tracking loops restarted in between,
00174     // and the TOW might have stepped if any of the two frame synchronizations
00175     // was false.
00176     static bool tracking_samples_are_consecutive(double rx_time_early_s,
00177         uint32_t tow_early_ms,
00178         double rx_time_late_s,
00179         uint32_t tow_late_ms);
00180
00181     // Finds the largest group of channels reporting the same time, and returns
00182     // its size and its latest TOW. A channel can report a wrong TOW, for
00183     // instance after a false frame synchronization of its Telemetry Decoder,
00184     // and such errors are unrelated from one channel to another, whereas the
00185     // channels reporting the right TOW agree with each other within
00186     // MAX_TOW_SPREAD_MS. If several groups have the same size, the one with the
00187     // latest TOW is returned, and is_unique is false if they are separate
00188     // groups: there is no reason to prefer the one returned. tow_ms holds the
00189     // TOW of each of the channels with a valid TOW, in [0, one week).
00190     static uint32_t find_tow_consensus(const std::vector<uint32_t>& tow_ms,
00191         uint32_t& latest_tow_ms,
00192         bool& is_unique);
00193
00194     void msg_handler_pvt_to_observables(const pmt::pmt_t& msg);
00195     double compute_T_rx_s(const Gnss_Synchro& a) const;
00196     bool interp_trk_obs(Gnss_Synchro& interpolated_obs, uint32_t ch, uint64_t rx_clock) const;
00197     bool has_fresh_trk_data(uint32_t ch, uint64_t rx_clock) const;
00198     void update_TOW(const std::vector<Gnss_Synchro>& data);
00199     void set_T_rx_TOW_ms(uint32_t tow_ms);
00200     bool rx_time_is_on_hold() const;
00201     void compute_pranges(std::vector<Gnss_Synchro>& data) const;
00202     void smooth_pseudoranges(std::vector<Gnss_Synchro>& data);
00203     void detect_cycle_slips(std::vector<Gnss_Synchro>& data, uint64_t rx_clock);
00204
00205     void set_tag_timestamp_in_sdr_timeframe(const std::vector<Gnss_Synchro>& data, uint64_t rx_clock);
00206
00207     void propagate_sensor_data(const std::vector<Gnss_Synchro>& data);
00208
00209     int32_t save_matfile() const;
00210
00211     Obs_Conf d_conf;
00212
00213     std::unique_ptr<Gnss_circular_deque<Gnss_Synchro» d_gnss_synchro_history;  // Tracking observable
    history
00214
00215     boost::circular_buffer<uint64_t> d_Rx_clock_buffer;  // time history
00216
00217     std::vector<std::queue<GnssTime» d_SourceTagTimestamps;
00218     std::queue<GnssTime> d_TimeChannelTagTimestamps;
00219
00220     std::queue<gr::tag_t> d_sensor_data_tags;
00221     std::uint64_t d_trq_last_sample{0};
00222
00223     std::vector<bool> d_channel_last_pll_lock;
00224     std::vector<double> d_channel_last_pseudorange_smooth;
00225     std::vector<double> d_channel_last_carrier_phase_rads;
00226     std::vector<bool> d_channel_has_previous_observation;     // carrier phase continuity bookkeeping
00227     std::vector<bool> d_channel_phase_discontinuity_pending;  // reported by the tracking block
00228     std::vector<bool> d_channel_last_pll_180_locked;          // Costas loop ambiguity state
00229     std::vector<uint64_t> d_channel_last_valid_epoch;
00230     std::vector<uint32_t> d_channel_last_valid_prn;
00231     std::vector<Gnss_Synchro> d_last_trk_data;  // latest tracking data per channel, for the Monitor
00232
00233     std::string d_dump_filename;
00234
00235     std::ofstream d_dump_file;
00236
00237     double d_smooth_filter_M;
00238     double d_T_rx_step_s;
00239     double d_last_rx_clock_round20ms_error;
00240
00241     uint64_t d_epoch_counter{0};
```

```
00242
00243     uint32_t d_T_rx_TOW_ms;
00244     uint32_t d_T_rx_step_ms;
00245     uint32_t d_T_rx_contradicted_ms{0};     // time that the channels have been contradicting the rx
    time
00246     uint32_t d_T_rx_TOW_unconfirmed_ms{0};  // time that the rx time has been running, if no group of
    channels agreed on it
00247     uint32_t d_T_status_report_timer_ms;
00248     uint32_t d_T_time_report_timer_ms{0};
00249     uint32_t d_nchannels_in;
00250     uint32_t d_nchannels_out;
00251
00252     bool d_T_rx_TOW_set;               // rx time follow GPST
00253     bool d_T_rx_TOW_confirmed{false};  // a group of channels agreed on the rx time
00254     bool d_always_output_gs;
00255     bool d_dump;
00256     bool d_dump_mat;
00257 };
00258
00259 /** \} */
00260 /** \} */
00261 #endif  // GNSS_SDR_HYBRID_OBSERVABLES_GS_H
```

13.225 obs_conf.h File Reference

Class that contains all the configuration parameters for generic observables block.

```
#include <cstdint>
#include <string>
```

Classes

- class Obs_Conf

13.225.1 Detailed Description

Class that contains all the configuration parameters for generic observables block.

Author

Javier Arribas, 2020. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file obs_conf.h.

13.226 obs_conf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file obs_conf.h
00003  * \brief Class that contains all the configuration parameters for generic
00004  * observables block
00005  * \author Javier Arribas, 2020. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_OBS_CONF_H
00019 #define GNSS_SDR_OBS_CONF_H
00020
00021 #include <cstdint>
00022 #include <string>
00023
00024 /** \addtogroup Observables
00025  * \{ */
```

```
00026 /** \addtogroup Observables_libs observables_libs
00027  * Utilities for GNSS observables configuration.
00028  * \{ */
00029
00030 class Obs_Conf
00031 {
00032 public:
00033     Obs_Conf();
00034
00035     std::string dump_filename{"obs_dump.dat"};
00036     int32_t smoothing_factor{0};
00037     uint32_t nchannels_in{0U};
00038     uint32_t nchannels_out{0U};
00039     uint32_t observable_interval_ms{20U};
00040     bool enable_carrier_smoothing{false};
00041     bool always_output_gs{false};
00042     bool dump{false};
00043     bool dump_mat{false};
00044     bool enable_E6{true};
00045     bool enable_monitor{false};
00046 };
00047
00048 /** \} */
00049 /** \} */
00050 #endif  // GNSS_SDR_OBS_CONF_H
```

13.227 rtklib_pvt.h File Reference

Interface of a Position Velocity and Time computation block.

```
#include "gnss_synchro.h"
#include "pvt_interface.h"
#include "rtklib.h"
#include "rtklib_pvt_gs.h"
#include <gnuradio/gr_complex.h>
#include <gnuradio/runtime_types.h>
#include <cstddef>
#include <ctime>
#include <map>
#include <string>
```

Classes

- class Rtklib_Pvt

  This class implements a PvtInterface for the RTKLIB PVT block.

13.227.1 Detailed Description

Interface of a Position Velocity and Time computation block.

Author

Javier Arribas, 2017. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file rtklib_pvt.h.

13.228 rtklib_pvt.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_pvt.h
00003  * \brief Interface of a Position Velocity and Time computation block
00004  * \author Javier Arribas, 2017. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_RTKLIB_PVT_H
00019 #define GNSS_SDR_RTKLIB_PVT_H
00020
00021 #include "gnss_synchro.h"
00022 #include "pvt_interface.h"           // for PvtInterface
00023 #include "rtklib.h"                  // for rtk_t
00024 #include "rtklib_pvt_gs.h"           // for rtklib_pvt_gs_sptr
00025 #include <gnuradio/gr_complex.h>     // for gr_complex
00026 #include <gnuradio/runtime_types.h>  // for basic_block_sptr, top_block_sptr
00027 #include <cstddef>                   // for size_t
00028 #include <ctime>                     // for time_t
00029 #include <map>                       // for map
00030 #include <string>                    // for string
00031
00032 /** \addtogroup PVT
00033  * Computation of Position, Velocity and Time from GNSS observables.
00034  * \{ */
00035 /** \addtogroup PVT_adapters pvt_adapters
00036  * Wrap GNU Radio PVT solvers with a PvtInterface
00037  * \{ */
00038
00039 class ConfigurationInterface;
00040 class Galileo_Almanac;
00041 class Galileo_Ephemeris;
00042 class Gps_Almanac;
00043 class Gps_Ephemeris;
00044
00045 /*!
00046  * \brief This class implements a PvtInterface for the RTKLIB PVT block
00047  *
00048  * Global configuration options used:
00049  *
00050  * GNSS-SDR.observation_date - approximate date of the signal capture in YYYY-MM-DD or YYYY format,
      used to resolve the GPS mod-1024 week-number rollover when post-processing recorded files (empty:
      derive it from the system clock)
00051  * GNSS-SDR.pre_2009_file - deprecated, use GNSS-SDR.observation_date instead; flag indicating a file
      captured in the Aug 1999 - Apr 2019 week-number era (false)
00052  * GNSS-SDR.observable_interval_ms - (20)
00053  *
00054  * It supports the following configuration options:
00055  *
00056  *  .dump - (false)
00057  *  .dump_filename - ("./pvt.dat")
00058  *  .dump_mat - (true)
00059  *  .rtk_trace_level - debug level for the RTKLIB methods (0)
00060  *
00061  *  .output_rate_ms - (500)
00062  *                    Note that the actual rate is the least common multiple of this value and
      GNSS-SDR.observable_interval_ms
00063  *  .display_rate_ms - (500)
00064  *
00065  *  .flag_nmea_tty_port - (false)
00066  *  .nmea_dump_filename - ("./nmea_pvt.nmea")
00067  *  .nmea_dump_devname - ("/dev/tty1")
00068  *
00069  *  .rinex_version - (3) values 2 (2.11), 3 (3.02) or 4 (4.02), overridden by -RINEX_version=n.nn
      command line argument
00070  *  .rinexobs_rate_ms - rate at which RINEX observations are written (1000).  Note that
00071  *                      the actual rate is the least common multiple of this value and
00072  *                      .output_rate_ms
00073  *  .rinex_name - (-RINEX_name command-line argument)
00074  *
00075  *  .flag_rtcm_tty_port - (false)
00076  *  .rtcm_dump_devname - ("/dev/pts/1")
00077  *  .flag_rtcm_server - (false)
00078  *  .rtcm_tcp_port - (2101)
00079  *  .rtcm_station_id - (1234)
00080  * Output rates ... all values are LCM with the computed output rate (above)
00081  *  .rtcm_MT1019_rate_ms - (5000)
00082  *  .rtcm_MT1020_rate_ms - (5000)
00083  *  .rtcm_MT1045_rate_ms - (5000)
00084  *  .rtcm_MSM_rate_ms - (1000)
00085  *  .rtcm_MT1077_rate_ms - (.rtcm_MSM_rate_ms)
00086  *  .rtcm_MT1087_rate_ms - (.rtcm_MSM_rate_ms)
00087  *  .rtcm_MT1097_rate_ms - (.rtcm_MSM_rate_ms)
00088  *
00089  * Fixed-base RTK input through NTRIP (disabled by default):
00090  *  .ntrip_client_enabled - (false)
```

```
00091  *  .ntrip_caster_address - caster hostname or IPv4 address, without a scheme or port ("")
00092  *  .ntrip_caster_port - (2101)
00093  *  .ntrip_mountpoint - caster mountpoint ("")
00094  *  .ntrip_username - Basic-authentication username ("")
00095  *  .ntrip_password - Basic-authentication password ("")
00096  *  .ntrip_password_env - environment variable containing the password (""), mutually exclusive with
       .ntrip_password
00097  *  .ntrip_version - 2 prefers v2 with one compatible v1 retry; 1 forces legacy v1 (2)
00098  *  .ntrip_tls_enabled - TLS 1.2-or-newer transport with system-CA certificate and hostname
       verification (false)
00099  *  .ntrip_inactivity_timeout_ms - (10000)
00100  *  .ntrip_reconnect_interval_ms - (10000)
00101  *  .ntrip_max_correction_age_s - (5.0)
00102  *  .ntrip_send_gga - upload the rover position as NMEA GGA, required by VRS/nearest-station casters
       (true)
00103  *  .ntrip_gga_period_ms - period of the GGA upload (10000)
00104  *  .ntrip_station_id - expected RTCM station ID, or zero to accept any stream station (0)
00105  *  .ntrip_fallback_to_single - report SOLQ_SINGLE while fixed-base data is unavailable (true)
00106  * Supported rover channel sets, per system: GPS 1C alone, 1C+2S, or 1C+L5;
00107  * Galileo 1B alone or 1B+5X; BeiDou B1C alone; in any combination across
00108  * systems. num_bands is derived from the channel set, navigation_system must
00109  * match the enabled constellations, and positioning_mode must be Static or
00110  * Kinematic. The base must provide its position through RTCM 1005/1006 and
00111  * observations through legacy 1002/1004 or MSM messages; VRS/nearest-station
00112  * casters are supported through the periodic GGA upload.
00113  *
00114  *  .kml_rate_ms - (1000)
00115  *  .gpx_rate_ms - (1000)
00116  *  .geojson_rate_ms - (1000)
00117  *  .nmea_rate_ms - (1000)
00118  *
00119  *  .positioning_mode - The RTKLIB positioning mode. ("Single") Supported values are "Single",
00120  *                      "Static", "Kinematic", "PPP_Static" and "PPP_Kinematic". Unsupported modes
00121  *                      include DGPS/DGNSS, Moving Baseline, Fixed, and PPP-fixed
00122  *  .num_bands - number of frequencies to use, between 1 and 3. Default is based on the channels
       configured
00123  *  .elevation_mask - (15.0). Value must be in the range [0,90.0]
00124  *  .dynamics_model - (0) 0:none, 1:velocity, 2:acceleration
00125  *  .satellite_ephemeris - ("Broadcast"). Supported values are "Broadcast" and "SBAS".
00126  *                      "SBAS" applies the fast and long-term satellite corrections received
00127  *                      from an SBAS telemetry channel. Legacy EGNOS L1 corrections augment
00128  *                      GPS L1 observations; satellites without valid corrections are excluded
00129  *  .sbas_satellite - (0). SBAS PRN whose correction stream is used. Zero selects the first
00130  *                      received stream; set a PRN in [120,138] when tracking multiple GEOs
00131  *  .iono_model - ("OFF"). Supported values are "OFF", "Broadcast", "SBAS", "Iono-Free-LC",
00132  *                      "Estimate_STEC", "IONEX". Unsupported values include QZSS broadcast, QZSS
00133  *                      LEX, and SLANT TEC.
00134  *  .trop_model - ("OFF"). Supported values are "OFF", "Saastamoinen", "SBAS", "Estimate_ZTD", and
00135  *                      "Estimate_ZTD_Grad". Unsupported values include ZTD correction and ZTD+grad
00136  *                      correction
00137  *  .phwindup - phase windup correction for PPP modes (0)
00138  *  .reject_GPS_IIA - whether the GPS Block IIA satellites in eclipse are excluded (0). Only applies
       in PPP-* modes
00139  *  .raim_fde - whether RAIM (receiver autonomous integrity monitoring) FDE (fault detection and
       exclusion) is enabled (0)
00140  *  .earth_tide - (0)
00141  *  .navigation_system - mask of navigation systems to use. Default based on configured channels
00142  *                      0x01:GPS, 0x02:SBAS, 0x04:GLONASS, 0x08:Galileo, 0x10:QZSS, 0x20:BeiDou,
00143  *                      0x40:IRNS, 0x80:LEO
00144  *
00145  *  .AR_GPS - Ambiguity Resolution mode for GPS ("Continuous"). Supported values are "OFF",
00146  *                      "Continuous", "Instantaneous", "Fix-and-Hold", "PPP-AR". Unsupported values
00147  *                      include PPP-AR ILS, WLNL, and TCAR.
00148  *  .AR_GLO - Ambiguity Resolution mode for GLONASS (1). Value must be in the range [0,3].
       (0:off,1:on,2:auto cal,3:ext cal)
00149  *  .AR_DBS - Ambiguity Resolution Mode for BeiDou (1). Value must be in the range [0,1]. (0:off,1:on)
00150  *  .min_ratio_to_fix_ambiguity - (3.0)
00151  *  .min_lock_to_fix_ambiguity - (0)
00152  *  .min_elevation_to_fix_ambiguity - minimum elevation (deg) to fix integer ambiguity (0.0)
00153  *  .outage_reset_ambiguity - (5)
00154  *  .slip_threshold - (0.05)
00155  *  .threshold_reject_gdop - if GDOP is over this value, the observable is excluded (30.0)
00156  *  .threshold_reject_innovation - if innovation is over this value, the observable is excluded (30.0)
00157  *  .number_filter_iter - number of iterations for the estimation filter (1)
00158  *  .bias_0 - (30.0)
00159  *  .iono_0 - (0.03)
00160  *  .trop_0 - (0.3)
00161  *  .sigma_bias - process noise stddev of carrier-phase bias(ambiguity)(cycle/sqrt(s)) (1e-4)
00162  *  .sigma_iono - process noise stddev of vertical ionospheric delay per 10km baseline (m/sqrt(s))
       (1e-3)
00163  *  .sigma_trop - process noise stddev of zenith tropospheric delay (m/sqrt(s)) (1e-4)
00164  *  .sigma_acch - process noise stddev of the receiver acceleration horizontal component
       (m/s2/sqrt(s)) (1e-1)
00165  *  .sigma_accv - process noise stddev of the receiver acceleration vertical component (m/s2/sqrt(s))
       (1e-2)
00166  *  .sigma_pos - (0.0)
00167  *  .code_phase_error_ratio_l1 - (100.0)
```

```
00168  *  .code_phase_error_ratio_l2 - (100.0)
00169  *  .code_phase_error_ratio_l5 - (100.0)
00170  *  .carrier_phase_error_factor_a - (0.003)
00171  *  .carrier_phase_error_factor_b - (0.003)
00172  *
00173  *  .output_enabled - (true)
00174  *  .rinex_output_enabled - (.output_enabled)
00175  *  .gpx_output_enabled - (.output_enabled)
00176  *  .geojson_output_enabled - (.output_enabled)
00177  *  .kml_output_enabled - (.output_enabled)
00178  *  .xml_output_enabled - (.output_enabled)
00179  *  .nmea_output_enabled - (.output_enabled)
00180  *  .rtcm_output_enabled - (false)
00181
00182  *  .output_path - directory to which output files are written (".")
00183  *  .rinex_output_path - (.output_path)
00184  *  .gpx_output_path - (.output_path)
00185  *  .geojson_output_path - (.output_path)
00186  *  .kml_output_path - (.output_path)
00187  *  .xml_output_path - (.output_path)
00188  *  .nmea_output_path - (.output_path)
00189  *  .rtcm_output_path - (.output_path)
00190  *
00191  *  .enable_monitor - enable the PVT monitor (false)
00192  *  .monitor_client_addresses - ("127.0.0.1")
00193  *  .monitor_udp_port - DO NOT USE THE DEFAULT (1234)
00194  *  .enable_protobuf - serialize using protocol buffers (true). Monitor.enable_protobuf if true, sets
     this to true
00195  *
00196  *  .enable_monitor_ephemeris - enable the ephemeris monitor (false)
00197  *  .monitor_ephemeris_client_addresses - ("127.0.0.1")
00198  *  .monitor_ephemeris_udp_port - DO NOT USE THE DEFAULT (1234)
00199  *
00200  *  .show_local_time_zone - (false)
00201  *  .enable_rx_clock_correction - (false)
00202  *  .max_clock_offset_ms - (40)
00203  */
00204 class Rtklib_Pvt : public PvtInterface
00205 {
00206 public:
00207     Rtklib_Pvt(const ConfigurationInterface* configuration,
00208         const std::string& role,
00209         unsigned int in_streams,
00210         unsigned int out_streams);
00211
00212     virtual ~Rtklib_Pvt();
00213
00214     inline std::string role() override
00215     {
00216         return role_;
00217     }
00218
00219     //!  Returns "RTKLIB_PVT"
00220     inline std::string implementation() override
00221     {
00222         return "RTKLIB_PVT";
00223     }
00224
00225     void clear_ephemeris() override;
00226     std::map<int, Gps_CNAV_Ephemeris> get_gps_cnav_ephemeris() const override;
00227     std::map<int, Glonass_Gnav_Ephemeris> get_glonass_ephemeris() const override;
00228     std::map<int, Glonass_Gnav_Almanac> get_glonass_almanac() const override;
00229     Glonass_Gnav_Utc_Model get_glonass_utc_model() const override;
00230     void clear_ephemeris_keep_almanac() override;
00231     std::map<int, Gps_Ephemeris> get_gps_ephemeris() const override;
00232     std::map<int, Galileo_Ephemeris> get_galileo_ephemeris() const override;
00233     std::map<int, Beidou_Dnav_Ephemeris> get_beidou_dnav_ephemeris() const override;
00234     std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav1_ephemeris() const override;
00235     std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav2_ephemeris() const override;
00236     std::map<int, Gps_Almanac> get_gps_almanac() const override;
00237     std::map<int, Galileo_Almanac> get_galileo_almanac() const override;
00238     std::map<int, Beidou_Dnav_Almanac> get_beidou_dnav_almanac() const override;
00239
00240     void connect(gr::top_block_sptr top_block) override;
00241     void disconnect(gr::top_block_sptr top_block) override;
00242     gr::basic_block_sptr get_left_block() override;
00243     gr::basic_block_sptr get_right_block() override;
00244
00245     inline void reset() override
00246     {
00247         return;
00248     }
00249
00250     //! All blocks must have an item_size() function implementation
00251     inline size_t item_size() override
00252     {
00253         return sizeof(Gnss_Synchro);
```

```
00254     }
00255
00256     bool get_latest_PVT(double* longitude_deg,
00257         double* latitude_deg,
00258         double* height_m,
00259         double* ground_speed_kmh,
00260         double* course_over_ground_deg,
00261         time_t* UTC_time) override;
00262
00263 private:
00264     rtklib_pvt_gs_sptr pvt_;
00265     rtk_t rtk{};
00266     std::string role_;
00267     unsigned int in_streams_;
00268     unsigned int out_streams_;
00269 };
00270
00271 /** \} */
00272 /** \} */
00273 #endif  // GNSS_SDR_RTKLIB_PVT_H
```

13.229 rtklib_pvt_gs.h File Reference

Interface of a Position Velocity and Time computation block.

```
#include "gnss_block_interface.h"
#include "gnss_sdr_make_unique.h"
#include "gnss_synchro.h"
#include "gnss_time.h"
#include "osnma_data.h"
#include "rtklib.h"
#include "sensor_data/sensor_data_aggregator.h"
#include "sensor_data/sensor_data_source_configuration.h"
#include <boost/date_time/gregorian/gregorian.hpp>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <boost/interprocess/ipc/message_queue.hpp>
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <atomic>
#include <chrono>
#include <cstddef>
#include <cstdint>
#include <ctime>
#include <fstream>
#include <map>
#include <memory>
#include <mutex>
#include <queue>
#include <string>
#include <vector>
```

Classes

- class rtklib_pvt_gs

  This class implements a block that computes the PVT solution using the RTKLIB integrated library.

Typedefs

- using rtklib_pvt_gs_sptr = gnss_shared_ptr<rtklib_pvt_gs>

Functions

- rtklib_pvt_gs_sptr **rtklib_make_pvt_gs** (uint32_t nchannels, const Pvt_Conf &conf_, const rtk_t &rtk, const SensorDataSourceConfiguration &sensor_data_configuration)

13.229.1 Detailed Description

Interface of a Position Velocity and Time computation block.

Author

Javier Arribas, 2017. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file rtklib_pvt_gs.h.

13.230 rtklib_pvt_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_pvt_gs.h
00003  * \brief Interface of a Position Velocity and Time computation block
00004  * \author Javier Arribas, 2017. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_RTKLIB_PVT_GS_H
00018 #define GNSS_SDR_RTKLIB_PVT_GS_H
00019
00020 #include "gnss_block_interface.h"
00021 #include "gnss_sdr_make_unique.h"
00022 #include "gnss_synchro.h"
00023 #include "gnss_time.h"
00024 #include "osnma_data.h"
00025 #include "rtklib.h"
00026 #include "sensor_data/sensor_data_aggregator.h"
00027 #include "sensor_data/sensor_data_source_configuration.h"
00028 #include <boost/date_time/gregorian/gregorian.hpp>
00029 #include <boost/date_time/posix_time/posix_time.hpp>
00030 #include <boost/interprocess/ipc/message_queue.hpp>
00031 #include <gnuradio/sync_block.h>  // for sync_block
00032 #include <gnuradio/types.h>       // for gr_vector_const_void_star
00033 #include <pmt/pmt.h>              // for pmt_t
00034 #include <atomic>                 // for atomic
00035 #include <chrono>                 // for system_clock
00036 #include <cstddef>                // for size_t
00037 #include <cstdint>                // for int32_t
00038 #include <ctime>                  // for time_t
00039 #include <fstream>                // for std::fstream
00040 #include <map>                    // for map
00041 #include <memory>                 // for shared_ptr, unique_ptr
00042 #include <mutex>                  // for mutex
00043 #include <queue>                  // for std::queue
00044 #include <string>                 // for string
00045 #include <vector>                 // for vector
00046
00047 /** \addtogroup PVT
00048  * \{ */
00049 /** \addtogroup PVT_gnuradio_blocks pvt_gr_blocks
00050  * GNU Radio blocks for the computation of PVT solutions.
00051  * \{ */
00052
00053
00054 class Gps_CNAV_Ephemeris;
00055 class Glonass_Gnav_Ephemeris;
00056 class Glonass_Gnav_Almanac;
00057 class Glonass_Gnav_Utc_Model;
00058 class Beidou_Dnav_Almanac;
00059 class Beidou_Dnav_Ephemeris;
```

```
00060 class Beidou_Cnav1_Ephemeris;
00061 class Galileo_Almanac;
00062 class Galileo_Ephemeris;
00063 class Galileo_HAS_data;
00064 class Galileo_Reduced_CED;
00065 class Geohash;
00066 class GeoJSON_Printer;
00067 class Glonass_Gnav_Ephemeris;
00068 class Gps_Almanac;
00069 class Gps_Ephemeris;
00070 class Gpx_Printer;
00071 class Kml_Printer;
00072 class Monitor_Pvt_Udp_Sink;
00073 class Monitor_Ephemeris_Udp_Sink;
00074 class Nmea_Printer;
00075 class Ntrip_Rtcm_Client;
00076 struct Ntrip_Rtcm_Snapshot;
00077 class Pvt_Conf;
00078 class Rinex_Printer;
00079 class Rtcm_Printer;
00080 class An_Packet_Printer;
00081 class Has_Simple_Printer;
00082 class Rtklib_Solver;
00083 class rtklib_pvt_gs;
00084
00085 using rtklib_pvt_gs_sptr = gnss_shared_ptr<rtklib_pvt_gs>;
00086
00087 rtklib_pvt_gs_sptr rtklib_make_pvt_gs(uint32_t nchannels,
00088     const Pvt_Conf& conf_,
00089     const rtk_t& rtk,
00090     const SensorDataSourceConfiguration& sensor_data_configuration);
00091
00092 /*!
00093  * \brief This class implements a block that computes the PVT solution using the RTKLIB integrated
      library
00094  */
00095 class rtklib_pvt_gs : public gr::sync_block
00096 {
00097 public:
00098     ~rtklib_pvt_gs() override;  //!< Default destructor
00099
00100     /*!
00101      * \brief Get latest set of GPS ephemeris from PVT block
00102      */
00103     std::map<int, Gps_CNAV_Ephemeris> get_gps_cnav_ephemeris_map() const;
00104     std::map<int, Glonass_Gnav_Ephemeris> get_glonass_ephemeris_map() const;
00105     std::map<int, Glonass_Gnav_Almanac> get_glonass_almanac_map() const;
00106     Glonass_Gnav_Utc_Model get_glonass_utc_model() const;
00107     std::map<int, Gps_Ephemeris> get_gps_ephemeris_map() const;
00108
00109     /*!
00110      * \brief Get latest set of GPS almanac from PVT block
00111      */
00112     std::map<int, Gps_Almanac> get_gps_almanac_map() const;
00113
00114     /*!
00115      * \brief Get latest set of Galileo ephemeris from PVT block
00116      */
00117     std::map<int, Galileo_Ephemeris> get_galileo_ephemeris_map() const;
00118
00119     /*!
00120      * \brief Get latest set of Galileo almanac from PVT block
00121      */
00122     std::map<int, Galileo_Almanac> get_galileo_almanac_map() const;
00123
00124     /*!
00125      * \brief Get latest set of BeiDou DNAV ephemeris from PVT block
00126      */
00127     std::map<int, Beidou_Dnav_Ephemeris> get_beidou_dnav_ephemeris_map() const;
00128     std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav1_ephemeris_map() const;
00129     std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav2_ephemeris_map() const;
00130
00131     /*!
00132      * \brief Get latest set of BeiDou DNAV almanac from PVT block
00133      */
00134     std::map<int, Beidou_Dnav_Almanac> get_beidou_dnav_almanac_map() const;
00135
00136     /*!
00137      * \brief Clears the published navigation snapshot now and the solver maps
00138      * at the worker's next work() or telemetry callback.
00139      */
00140     void clear_ephemeris();
00141
00142     /*!
00143      * \brief Clears the ephemeris entries of the published navigation snapshot
00144      * now and the solver ephemeris maps at the worker's next work() or telemetry
00145      * callback, leaving the GPS, Galileo, BeiDou and GLONASS almanacs untouched.
```

```
00146      */
00147     void clear_ephemeris_keep_almanac();
00148
00149     /*!
00150      * \brief Interpolates one observable between two epochs, in the carrier
00151      * polarity frame of the later epoch.
00152      *
00153      * A plain linear carrier-phase interpolation across a PLL-180 half-cycle
00154      * re-resolution would leak a fraction of the pi step into the output with
00155      * no slip flag attached; the early phase is therefore shifted into the
00156      * late epoch's polarity frame first, the late polarity is stamped on the
00157      * output, and a slip on either endpoint (or a polarity change itself)
00158      * marks the output.
00159      */
00160     static Gnss_Synchro interpolate_observable(const Gnss_Synchro& early,
00161         const Gnss_Synchro& late,
00162         double time_factor,
00163         double rx_time_s);
00164
00165     /*!
00166      * \brief Get the latest Position WGS84 [deg], Ground Velocity, Course over Ground, and UTC Time,
    if available
00167      */
00168     bool get_latest_PVT(double* longitude_deg,
00169         double* latitude_deg,
00170         double* height_m,
00171         double* ground_speed_kmh,
00172         double* course_over_ground_deg,
00173         time_t* UTC_time) const;
00174
00175     int work(int noutput_items, gr_vector_const_void_star& input_items,
00176         gr_vector_void_star& output_items) override;  //!< PVT Signal Processing
00177
00178 private:
00179     friend rtklib_pvt_gs_sptr rtklib_make_pvt_gs(uint32_t nchannels,
00180         const Pvt_Conf& conf_,
00181         const rtk_t& rtk,
00182         const SensorDataSourceConfiguration& sensor_data_configuration);
00183
00184     rtklib_pvt_gs(uint32_t nchannels,
00185         const Pvt_Conf& conf_,
00186         const rtk_t& rtk,
00187         const SensorDataSourceConfiguration& sensor_data_configuration);
00188
00189     void log_source_timetag_info(double RX_time_ns, double TAG_time_ns);
00190
00191     void msg_handler_telemetry(const pmt::pmt_t& msg);
00192
00193     void msg_handler_has_data(const pmt::pmt_t& msg);
00194
00195     void msg_handler_osnma(const pmt::pmt_t& msg);
00196
00197     void initialize_and_apply_carrier_phase_offset();
00198
00199     void apply_rx_clock_offset(std::map<int, Gnss_Synchro>& observables_map,
00200         double rx_clock_offset_s);
00201
00202     void update_HAS_corrections();
00203
00204     void report_fixed_base_status();
00205
00206     void report_solution_status();
00207
00208     void report_solution_outage();
00209
00210     // slip flags ride single observables epochs, but the user solver only
00211     // samples the maps on output epochs: every flag seen while an epoch
00212     // passes through t1 is latched per channel and consumed on the next
00213     // interpolated output, so no slip vanishes between outputs
00214     std::map<int, bool> d_interp_pending_cycle_slip;
00215     std::map<int, bool> d_interp_pending_half_cycle_slip;
00216
00217     std::map<int, Gnss_Synchro> interpolate_observables(const std::map<int, Gnss_Synchro>&
    observables_map_t0,
00218         const std::map<int, Gnss_Synchro>& observables_map_t1,
00219         double rx_time_s);
00220
00221     inline std::time_t convert_to_time_t(const boost::posix_time::ptime pt) const
00222     {
00223         return (pt - boost::posix_time::ptime(boost::gregorian::date(1970, 1, 1))).total_seconds();
00224     }
00225
00226     std::vector<std::string> split_string(const std::string& s, char delim) const;
00227
00228     bool send_ttff_msg(double ttff) const;
00229     bool save_gnss_synchro_map_xml(const std::string& file_name);  // debug helper function
00230     bool load_gnss_synchro_map_xml(const std::string& file_name);  // debug helper function
```

```
00231
00232     std::fstream d_log_timetag_file;
00233
00234     std::unique_ptr<SensorDataAggregator> d_sensor_data_aggregator;
00235
00236     enum class NavigationData
00237     {
00238         GpsCnavEphemeris,
00239         GlonassEphemeris,
00240         GlonassAlmanac,
00241         GlonassUtcModel,
00242         GpsEphemeris,
00243         GpsAlmanac,
00244         GalileoEphemeris,
00245         GalileoAlmanac,
00246         BeidouEphemeris,
00247         BeidouCnavEphemeris,
00248         BeidouAlmanac,
00249         RetainedAlmanacs
00250     };
00251     enum class NavigationClear : uint8_t
00252     {
00253         None,
00254         EphemerisOnly,
00255         All
00256     };
00257     struct NavigationSnapshot;
00258     struct LatestPvt
00259     {
00260         double longitude_deg = 0.0;
00261         double latitude_deg = 0.0;
00262         double height_m = 0.0;
00263         double ground_speed_kmh = 0.0;
00264         double course_over_ground_deg = 0.0;
00265         time_t utc_time = 0;
00266         bool valid = false;
00267     };
00268
00269     std::shared_ptr<const NavigationSnapshot> navigation_snapshot() const;
00270     void publish_navigation_snapshot(NavigationData data);
00271     void request_navigation_clear(NavigationClear kind);
00272     void apply_pending_navigation_clear();
00273     void publish_latest_pvt();
00274
00275     // Only pointer exchanges and the small PVT value copy hold this mutex.
00276     // Map construction, copying and destruction happen outside the lock.
00277     mutable std::mutex d_snapshot_mutex;
00278     const std::shared_ptr<const NavigationSnapshot> d_empty_navigation_snapshot;
00279     std::shared_ptr<const NavigationSnapshot> d_navigation_snapshot;
00280     LatestPvt d_latest_pvt;
00281     std::atomic<uint32_t> d_navigation_generation{0};
00282     NavigationClear d_pending_navigation_clear = NavigationClear::None;  // guarded by
    d_snapshot_mutex
00283     uint32_t d_applied_navigation_generation = 0;                        // GNU Radio worker only
00284     std::shared_ptr<Rtklib_Solver> d_internal_pvt_solver;
00285     std::shared_ptr<Rtklib_Solver> d_user_pvt_solver;
00286
00287     std::unique_ptr<boost::interprocess::message_queue> d_mq;
00288
00289     std::unique_ptr<Rinex_Printer> d_rp;
00290     std::unique_ptr<Kml_Printer> d_kml_dump;
00291     std::unique_ptr<Gpx_Printer> d_gpx_dump;
00292     std::unique_ptr<Nmea_Printer> d_nmea_printer;
00293     std::unique_ptr<Ntrip_Rtcm_Client> d_ntrip_client;
00294     // Cached copy of the client's latest snapshot, refreshed only when the
00295     // client's snapshot generation changes (the deep copy is expensive at the
00296     // observables rate).
00297     std::unique_ptr<Ntrip_Rtcm_Snapshot> d_fixed_base_snapshot;
00298     std::unique_ptr<GeoJSON_Printer> d_geojson_printer;
00299     std::unique_ptr<Rtcm_Printer> d_rtcm_printer;
00300     std::unique_ptr<Monitor_Pvt_Udp_Sink> d_udp_sink_ptr;
00301     std::unique_ptr<Monitor_Ephemeris_Udp_Sink> d_eph_udp_sink_ptr;
00302     std::unique_ptr<Has_Simple_Printer> d_has_simple_printer;
00303     std::unique_ptr<An_Packet_Printer> d_an_printer;
00304
00305     std::chrono::time_point<std::chrono::system_clock> d_start;
00306     std::chrono::time_point<std::chrono::system_clock> d_end;
00307
00308     const std::string d_queue_name;
00309     std::string d_dump_filename;
00310     std::string d_xml_base_path;
00311     std::string d_local_time_str;
00312
00313     std::vector<bool> d_channel_initialized;
00314     std::vector<double> d_initial_carrier_phase_offset_estimation_rads;
00315
00316     std::map<int, Gnss_Synchro> d_gnss_observables_map;
```

```
00317     std::map<int, Gnss_Synchro> d_gnss_observables_map_t0;
00318     std::map<int, Gnss_Synchro> d_gnss_observables_map_t1;
00319     std::map<uint32_t, std::map<uint32_t, uint64_t» d_auth_nav_data_map;
00320
00321     std::queue<GnssTime> d_TimeChannelTagTimestamps;
00322
00323     boost::posix_time::time_duration d_utc_diff_time;
00324     const std::unique_ptr<Geohash> d_geohash;
00325
00326     const size_t d_gps_ephemeris_sptr_type_hash_code;
00327     const size_t d_gps_iono_sptr_type_hash_code;
00328     const size_t d_gps_utc_model_sptr_type_hash_code;
00329     const size_t d_gps_cnav_ephemeris_sptr_type_hash_code;
00330     const size_t d_gps_cnav_eop_sptr_type_hash_code;
00331     const size_t d_gps_cnav_iono_sptr_type_hash_code;
00332     const size_t d_gps_cnav_utc_model_sptr_type_hash_code;
00333     const size_t d_gps_almanac_sptr_type_hash_code;
00334     const size_t d_qzss_iono_sptr_type_hash_code;
00335     const size_t d_qzss_utc_model_sptr_type_hash_code;
00336     const size_t d_qzss_cnav_eop_sptr_type_hash_code;
00337     const size_t d_qzss_cnav_iono_sptr_type_hash_code;
00338     const size_t d_qzss_cnav_utc_model_sptr_type_hash_code;
00339     const size_t d_galileo_ephemeris_sptr_type_hash_code;
00340     const size_t d_galileo_reduced_ced_sptr_type_hash_code;
00341     const size_t d_galileo_iono_sptr_type_hash_code;
00342     const size_t d_galileo_utc_model_sptr_type_hash_code;
00343     const size_t d_galileo_almanac_helper_sptr_type_hash_code;
00344     const size_t d_galileo_almanac_sptr_type_hash_code;
00345     const size_t d_glonass_gnav_ephemeris_sptr_type_hash_code;
00346     const size_t d_glonass_gnav_utc_model_sptr_type_hash_code;
00347     const size_t d_glonass_gnav_almanac_sptr_type_hash_code;
00348     const size_t d_beidou_dnav_ephemeris_sptr_type_hash_code;
00349     const size_t d_beidou_dnav_iono_sptr_type_hash_code;
00350     const size_t d_beidou_dnav_utc_model_sptr_type_hash_code;
00351     const size_t d_beidou_dnav_almanac_sptr_type_hash_code;
00352     const size_t d_beidou_cnav1_ephemeris_sptr_type_hash_code;
00353     const size_t d_beidou_cnav1_iono_sptr_type_hash_code;
00354     const size_t d_beidou_cnav1_utc_model_sptr_type_hash_code;
00355     const size_t d_beidou_cnav1_page_data_sptr_type_hash_code;
00356     const size_t d_sbas_raw_message_sptr_type_hash_code;
00357     const size_t d_galileo_has_data_sptr_type_hash_code;
00358
00359     const double d_rinex_version;
00360     double d_rx_time;
00361     uint64_t d_local_counter_ms;
00362     uint64_t d_timestamp_rx_clock_offset_correction_msg_ms;
00363
00364     const int32_t d_rinexobs_rate_ms;
00365     int32_t d_rtcm_MT1045_rate_ms;  // Galileo Broadcast Ephemeris
00366     int32_t d_rtcm_MT1019_rate_ms;  // GPS Broadcast Ephemeris (orbits)
00367     int32_t d_rtcm_MT1020_rate_ms;  // GLONASS Broadcast Ephemeris (orbits)
00368     int32_t d_rtcm_MT1077_rate_ms;  // The type 7 Multiple Signal Message format for the USA's GPS
    system, popular
00369     int32_t d_rtcm_MT1087_rate_ms;  // GLONASS MSM7. The type 7 Multiple Signal Message format for the
    Russian GLONASS system
00370     int32_t d_rtcm_MT1097_rate_ms;  // Galileo MSM7. The type 7 Multiple Signal Message format for
    Europe's Galileo system
00371     int32_t d_rtcm_MSM_rate_ms;
00372     const int32_t d_kml_rate_ms;
00373     const int32_t d_gpx_rate_ms;
00374     const int32_t d_geojson_rate_ms;
00375     const int32_t d_nmea_rate_ms;
00376     const int32_t d_an_rate_ms;
00377     const int32_t d_output_rate_ms;
00378     const int32_t d_display_rate_ms;
00379     const int32_t d_report_rate_ms;
00380     const int32_t d_max_obs_block_rx_clock_offset_ms;
00381
00382     const uint32_t d_nchannels;
00383     const uint32_t d_signal_enabled_flags;
00384     const uint32_t d_observable_interval_ms;
00385     const double d_ntrip_max_correction_age_s;
00386     uint32_t d_pvt_errors_counter;         // consecutive epochs without a solution, whatever the
    reason
00387     uint32_t d_pvt_solver_errors_counter;  // epochs in which the solver tried and failed since the
    last solution, up to 100
00388     int d_last_fixed_base_status;
00389     uint64_t d_ntrip_snapshot_generation = 0;
00390     int d_last_solution_status;
00391
00392     bool d_dump;
00393     const bool d_dump_mat;
00394     const bool d_rinex_output_enabled;
00395     bool d_geojson_output_enabled;
00396     bool d_gpx_output_enabled;
00397     bool d_kml_output_enabled;
00398     bool d_nmea_output_file_enabled;
```

```
00399     bool d_rtcm_enabled;
00400     bool d_first_fix;
00401     bool d_xml_storage;
00402     const bool d_flag_monitor_pvt_enabled;
00403     const bool d_flag_monitor_ephemeris_enabled;
00404     const bool d_show_local_time_zone;
00405     const bool d_enable_rx_clock_correction;
00406     const bool d_ntrip_client_enabled;
00407     bool d_enable_has_messages;
00408     const bool d_an_printer_enabled;
00409     bool d_log_timetag;
00410     const bool d_use_has_corrections;
00411     const bool d_use_unhealthy_sats;
00412     const bool d_osnma_strict;
00413 };
00414
00415
00416 /** \} */
00417 /** \} */
00418 #endif  // GNSS_SDR_RTKLIB_PVT_GS_H
```

13.231 an_packet_printer.h File Reference

Interface of a class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.

```
#include "gnss_synchro.h"
#include <array>
#include <chrono>
#include <cstddef>
#include <cstdint>
#include <map>
#include <string>
```

Classes

- struct sdr_gnss_packet_t
- struct an_packet_t
- class An_Packet_Printer

  class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.

13.231.1 Detailed Description

Interface of a class that prints PVT solutions in a serial device following a custom version of the Advanced Navigation Packet Protocol.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

Miguel Angel Gomez Lopez, 2021. gomezlma(at)inta.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file an_packet_printer.h.

13.232 an_packet_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file an_packet_printer.h
00003  * \brief Interface of a class that prints PVT solutions in a serial device
00004  * following a custom version of the Advanced Navigation Packet Protocol
00005  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00006  * \author Miguel Angel Gomez Lopez, 2021. gomezlma(at)inta.es
00007  *
00008  *
```

```
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020
00021 #ifndef GNSS_SDR_AN_PACKET_PRINTER_H
00022 #define GNSS_SDR_AN_PACKET_PRINTER_H
00023
00024 #include "gnss_synchro.h"
00025 #include <array>
00026 #include <chrono>
00027 #include <cstddef>
00028 #include <cstdint>
00029 #include <map>
00030 #include <string>
00031
00032 /** \addtogroup PVT
00033  * \{ */
00034 /** \addtogroup PVT_libs
00035  * \{ */
00036
00037 class Rtklib_Solver;
00038
00039 struct sdr_gnss_packet_t
00040 {
00041     uint8_t nsvfix;              // number of sats used in PVT fix
00042     uint8_t gps_satellites;      // number of tracked GPS satellites
00043     uint8_t galileo_satellites;  // number of tracked Galileo satellites
00044     uint32_t microseconds;       // from start of receiver operation
00045     double latitude;             // in [rad]
00046     double longitude;            // in [rad]
00047     double height;               // in [m]
00048     float velocity[3];           // North, East, Down, in [m/s]
00049
00050     struct
00051     {
00052         uint8_t prn;      // PRN ID. Galileo sats expressed as PRN + 100
00053         uint8_t snr;      // in [dB-Hz]
00054         int16_t doppler;  // in [Hz], saturates at +32767 / -32768 Hz
00055     } sats[16];
00056
00057     uint32_t reserved;
00058     uint16_t status;
00059 };
00060
00061
00062 struct an_packet_t
00063 {
00064     uint8_t header[4];
00065     uint8_t data[113];  // SDR_GNSS_PACKET_LENGTH
00066 };
00067
00068
00069 /*!
00070  * \brief class that prints PVT solutions in a serial device following a custom
00071  * version of the Advanced Navigation Packet Protocol.
00072  */
00073 class An_Packet_Printer
00074 {
00075 public:
00076     /*!
00077      * \brief Default constructor.
00078      */
00079     explicit An_Packet_Printer(const std::string& an_dump_devname);
00080
00081     /*!
00082      * \brief Default destructor.
00083      */
00084     ~An_Packet_Printer();
00085
00086     /*!
00087      * \brief Print AN packet to the initialized device.
00088      */
00089     bool print_packet(const Rtklib_Solver* const pvt_data, const std::map<int, Gnss_Synchro>&
    gnss_observables_map);
00090
00091     /*!
00092      * \brief Close serial port. Also done in the destructor, this is only
00093      * for testing.
00094      */
```

```
00095     void close_serial() const;
00096
00097 private:
00098     const std::array<uint16_t, 256> d_crc16_table = {
00099         {0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7, 0x8108, 0x9129, 0xa14a,
      0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef, 0x1231, 0x0210, 0x3273,
00100             0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6, 0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c,
      0xf3ff, 0xe3de, 0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485, 0xa56a, 0xb54b,
      0x8528,
00101             0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d, 0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6,
      0x5695, 0x46b4, 0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc, 0x48c4, 0x58e5,
      0x6886,
00102             0x78a7, 0x0840, 0x1861, 0x2802, 0x3823, 0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969,
      0xa90a, 0xb92b, 0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12, 0xdbfd, 0xcbdc,
      0xfbbf,
00103             0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a, 0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03,
      0x0c60, 0x1c41, 0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49, 0x7e97, 0x6eb6,
      0x5ed5,
00104             0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70, 0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a,
      0x9f59, 0x8f78, 0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f, 0x1080, 0x00a1,
      0x30c2,
00105             0x20e3, 0x5004, 0x4025, 0x7046, 0x6067, 0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c,
      0xe37f, 0xf35e, 0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256, 0xb5ea, 0xa5cb,
      0x95a8,
00106             0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d, 0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447,
      0x5424, 0x4405, 0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c, 0x26d3, 0x36f2,
      0x0691,
00107             0x16b0, 0x6657, 0x7676, 0x4615, 0x5634, 0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9,
      0xb98a, 0xa9ab, 0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3, 0xcb7d, 0xdb5c,
      0xeb3f,
00108             0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a, 0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0,
      0x2ab3, 0x3a92, 0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9, 0x7c26, 0x6c07,
      0x5c64,
00109             0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1, 0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba,
      0x8fd9, 0x9ff8, 0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0}};
00110
00111     const size_t SDR_GNSS_PACKET_LENGTH = 113;
00112     const uint8_t SDR_GNSS_PACKET_ID = 201;
00113
00114     int init_serial(const std::string& serial_device);
00115     void update_sdr_gnss_packet(sdr_gnss_packet_t* _packet, const Rtklib_Solver* const pvt, const
      std::map<int, Gnss_Synchro>& gnss_observables_map) const;
00116     void encode_gnss_cttc_packet(sdr_gnss_packet_t* sdr_gnss_packet, an_packet_t* _packet) const;
00117     uint16_t calculate_crc16(const void* data, uint16_t length) const;
00118     uint8_t calculate_header_lrc(const uint8_t* data) const;
00119     void an_packet_encode(an_packet_t* an_packet) const;
00120     void encode_sdr_gnss_packet(sdr_gnss_packet_t* sdr_gnss_packet, an_packet_t* _packet) const;
00121     void LSB_bytes_to_array(void* _in, int offset, uint8_t* _out, uint8_t var_size) const;
00122
00123     std::chrono::time_point<std::chrono::system_clock> d_start;
00124     std::string d_an_devname;
00125     int d_an_dev_descriptor;  // serial device descriptor (i.e. COM port)
00126 };
00127
00128 /** \} */
00129 /** \} */
00130 #endif  // GNSS_SDR_AN_PACKET_PRINTER_H
```

13.233 galileo_ephemeris_store.h File Reference

Source-aware storage for Galileo I/NAV and F/NAV ephemerides.

```
#include "galileo_ephemeris.h"
#include <map>
```

Classes

- class Galileo_Ephemeris_Store

13.233.1 Detailed Description

Source-aware storage for Galileo I/NAV and F/NAV ephemerides.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_ephemeris_store.h.

13.234 galileo_ephemeris_store.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_ephemeris_store.h
00003  * \brief Source-aware storage for Galileo I/NAV and F/NAV ephemerides
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GALILEO_EPHEMERIS_STORE_H
00018 #define GNSS_SDR_GALILEO_EPHEMERIS_STORE_H
00019
00020 #include "galileo_ephemeris.h"
00021 #include <map>
00022
00023
00024 class Galileo_Ephemeris_Store
00025 {
00026 public:
00027     using Ephemeris_Map = std::map<int, Galileo_Ephemeris>;
00028
00029     bool insert(const Galileo_Ephemeris& ephemeris)
00030     {
00031         Ephemeris_Map* destination = mutable_map(ephemeris.nav_message_type);
00032         if (destination == nullptr)
00033             {
00034                 return false;
00035             }
00036         (*destination)[static_cast<int>(ephemeris.PRN)] = ephemeris;
00037         return true;
00038     }
00039
00040     const Galileo_Ephemeris* find(int prn, Galileo_Nav_Message_Type source) const
00041     {
00042         const Ephemeris_Map* source_map = map(source);
00043         if (source_map == nullptr)
00044             {
00045                 return nullptr;
00046             }
00047         const auto ephemeris = source_map->find(prn);
00048         return ephemeris == source_map->cend() ? nullptr : &ephemeris->second;
00049     }
00050
00051     const Ephemeris_Map& inav() const
00052     {
00053         return d_inav_ephemeris_map;
00054     }
00055
00056     Ephemeris_Map& inav()
00057     {
00058         return d_inav_ephemeris_map;
00059     }
00060
00061     const Ephemeris_Map& fnav() const
00062     {
00063         return d_fnav_ephemeris_map;
00064     }
00065
00066     Ephemeris_Map& fnav()
00067     {
00068         return d_fnav_ephemeris_map;
00069     }
00070
00071     const Ephemeris_Map& by_source(Galileo_Nav_Message_Type source) const
```

```
00072     {
00073         static const Ephemeris_Map empty_ephemeris_map;
00074         const Ephemeris_Map* source_map = map(source);
00075         return source_map == nullptr ? empty_ephemeris_map : *source_map;
00076     }
00077
00078     Ephemeris_Map combined_view(Galileo_Nav_Message_Type preferred_source) const
00079     {
00080         Ephemeris_Map result;
00081         const Ephemeris_Map& secondary = preferred_source == Galileo_Nav_Message_Type::FNAV ?
      d_inav_ephemeris_map : d_fnav_ephemeris_map;
00082         const Ephemeris_Map& preferred = preferred_source == Galileo_Nav_Message_Type::FNAV ?
      d_fnav_ephemeris_map : d_inav_ephemeris_map;
00083         result.insert(secondary.cbegin(), secondary.cend());
00084         for (const auto& ephemeris : preferred)
00085             {
00086                 Galileo_Ephemeris merged = ephemeris.second;
00087                 // F/NAV never carries E1B_HS/E1B_DVS/E5b_HS/E5b_DVS (I/NAV-only),
00088                 // and vice versa for E5a_HS/E5a_DVS -- those default to "0 == OK"
00089                 // instead of the real value, so fill them in from the secondary
00090                 // source when available rather than leaving the default.
00091                 const auto secondary_it = secondary.find(ephemeris.first);
00092                 if (secondary_it != secondary.cend())
00093                     {
00094                         if (preferred_source == Galileo_Nav_Message_Type::FNAV)
00095                             {
00096                                 merged.E1B_HS = secondary_it->second.E1B_HS;
00097                                 merged.E1B_DVS = secondary_it->second.E1B_DVS;
00098                                 merged.E5b_HS = secondary_it->second.E5b_HS;
00099                                 merged.E5b_DVS = secondary_it->second.E5b_DVS;
00100                             }
00101                         else
00102                             {
00103                                 merged.E5a_HS = secondary_it->second.E5a_HS;
00104                                 merged.E5a_DVS = secondary_it->second.E5a_DVS;
00105                             }
00106                     }
00107                 result[ephemeris.first] = merged;
00108             }
00109         return result;
00110     }
00111
00112     bool empty() const
00113     {
00114         return d_inav_ephemeris_map.empty() && d_fnav_ephemeris_map.empty();
00115     }
00116
00117     void clear()
00118     {
00119         d_inav_ephemeris_map.clear();
00120         d_fnav_ephemeris_map.clear();
00121     }
00122
00123 private:
00124     const Ephemeris_Map* map(Galileo_Nav_Message_Type source) const
00125     {
00126         if (source == Galileo_Nav_Message_Type::INAV)
00127             {
00128                 return &d_inav_ephemeris_map;
00129             }
00130         if (source == Galileo_Nav_Message_Type::FNAV)
00131             {
00132                 return &d_fnav_ephemeris_map;
00133             }
00134         return nullptr;
00135     }
00136
00137     Ephemeris_Map* mutable_map(Galileo_Nav_Message_Type source)
00138     {
00139         if (source == Galileo_Nav_Message_Type::INAV)
00140             {
00141                 return &d_inav_ephemeris_map;
00142             }
00143         if (source == Galileo_Nav_Message_Type::FNAV)
00144             {
00145                 return &d_fnav_ephemeris_map;
00146             }
00147         return nullptr;
00148     }
00149
00150     Ephemeris_Map d_inav_ephemeris_map;
00151     Ephemeris_Map d_fnav_ephemeris_map;
00152 };
00153
00154 #endif  // GNSS_SDR_GALILEO_EPHEMERIS_STORE_H
```

13.235 geohash.h File Reference

Interface of a class that encodes / decodes geohashes.

```
#include <array>
#include <string>
```

Classes

- class Geohash

  Class for geohash encoding / decoding See https://en.wikipedia.org/wiki/Geohash.

13.235.1 Detailed Description

Interface of a class that encodes / decodes geohashes.

Author

Carles Fernandez-Prades, 2023. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file geohash.h.

13.236 geohash.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file geohash.h
00003  * \brief Interface of a class that encodes / decodes geohashes
00004  * \author Carles Fernandez-Prades, 2023. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GEOHASH_H
00020 #define GNSS_SDR_GEOHASH_H
00021
00022 #include <array>
00023 #include <string>
00024
00025 /** \addtogroup PVT
00026  * \{ */
00027 /** \addtogroup PVT_libs
00028  * \{ */
00029
00030 /*!
00031  * \brief Class for geohash encoding / decoding
00032  * See https://en.wikipedia.org/wiki/Geohash
00033  */
00034 class Geohash
00035 {
00036 public:
00037     Geohash() = default;
00038
00039     /**
00040      * Encodes latitude/longitude to geohash, either to specified precision or
00041      * to automatically evaluated precision.
00042      *
00043      * @param   {double} lat - Latitude in degrees.
00044      * @param   {double} lon - Longitude in degrees.
00045      * @param   {int} [precision] - Number of characters in resulting geohash.
00046      * @returns {string} Geohash of supplied latitude/longitude.
00047      * @throws  Invalid geohash.
00048      *
```

```
00049      */
00050     std::string encode(double lat, double lon, int precision = -1) const;
00051
00052     /**
00053      * Decode geohash to latitude/longitude (location is approximate centre of
00054      * geohash cell, to reasonable precision).
00055      *
00056      * @param   {string} geohash - Geohash string to be converted to
00057      * latitude/longitude.
00058      * @returns {lat, lon} (Center of) geohashed location.
00059      * @throws  Invalid geohash.
00060      *
00061      */
00062     std::array<double, 2> decode(std::string geohash) const;
00063
00064 private:
00065     /*
00066      * Returns SW/NE latitude/longitude bounds of specified geohash.
00067      */
00068     std::array<double, 4> bounds(std::string geohash) const;
00069     std::string base32{"0123456789bcdefghjkmnpqrstuvwxyz"};
00070 };
00071
00072 /** \} */
00073 /** \} */
00074 #endif  // GNSS_SDR_GEOHASH_H
```

13.237 geojson_printer.h File Reference

Interface of a class that prints PVT solutions in GeoJSON format.

```
#include <fstream>
#include <string>
```

Classes

- class GeoJSON_Printer

  Prints PVT solutions in GeoJSON format file.

13.237.1 Detailed Description

Interface of a class that prints PVT solutions in GeoJSON format.

Author

Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file geojson_printer.h.

13.238 geojson_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file geojson_printer.h
00003  * \brief Interface of a class that prints PVT solutions in GeoJSON format
00004  * \author Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GEOJSON_PRINTER_H
```

```
00020 #define GNSS_SDR_GEOJSON_PRINTER_H
00021
00022
00023 #include <fstream>
00024 #include <string>
00025
00026 /** \addtogroup PVT
00027  * \{ */
00028 /** \addtogroup PVT_libs
00029  * \{ */
00030
00031
00032 class Pvt_Solution;
00033
00034 /*!
00035  * \brief Prints PVT solutions in GeoJSON format file
00036  *
00037  * See https://tools.ietf.org/html/rfc7946
00038  */
00039 class GeoJSON_Printer
00040 {
00041 public:
00042     explicit GeoJSON_Printer(const std::string& base_path = ".");
00043     ~GeoJSON_Printer();
00044     bool set_headers(const std::string& filename, bool time_tag_name = true);
00045     bool print_position(const Pvt_Solution* const position);
00046     bool close_file();
00047
00048 private:
00049     std::ofstream geojson_file;
00050     std::string filename_;
00051     std::string geojson_base_path;
00052     bool first_pos;
00053 };
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_GEOJSON_PRINTER_H
```

13.239 gpx_printer.h File Reference

Interface of a class that prints PVT information to a gpx file.

```
#include <fstream>
#include <string>
```

Classes

- class Gpx_Printer

  Prints PVT information to GPX format file.

13.239.1 Detailed Description

Interface of a class that prints PVT information to a gpx file.

Author

Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gpx_printer.h.

13.240 gpx_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gpx_printer.h
00003  * \brief Interface of a class that prints PVT information to a gpx file
00004  * \author Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00005  *
00006  *
```

```
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GPX_PRINTER_H
00020 #define GNSS_SDR_GPX_PRINTER_H
00021
00022
00023 #include <fstream>
00024 #include <string>
00025
00026 /** \addtogroup PVT
00027  * \{ */
00028 /** \addtogroup PVT_libs
00029  * \{ */
00030
00031
00032 class Pvt_Solution;
00033
00034 /*!
00035  * \brief Prints PVT information to GPX format file
00036  *
00037  * See https://www.topografix.com/gpx.asp
00038  */
00039 class Gpx_Printer
00040 {
00041 public:
00042     explicit Gpx_Printer(const std::string& base_path = ".");
00043     ~Gpx_Printer() noexcept;
00044     bool set_headers(const std::string& filename, bool time_tag_name = true);
00045     bool print_position(const Pvt_Solution* const position);
00046     bool close_file();
00047
00048 private:
00049     std::ofstream gpx_file;
00050     std::string gpx_filename;
00051     std::string indent;
00052     std::string gpx_base_path;
00053     bool positions_printed;
00054 };
00055
00056
00057 /** \} */
00058 /** \} */
00059 #endif  // GNSS_SDR_GPX_PRINTER_H
```

13.241 has_simple_printer.h File Reference

Interface of a class that prints HAS messages content in a txt file.

```
#include <cstddef>
#include <fstream>
#include <mutex>
#include <string>
#include <vector>
```

Classes

- class Has_Simple_Printer

  Prints HAS messages content in a txt file. See HAS-SIS-ICD for a message description.

13.241.1 Detailed Description

Interface of a class that prints HAS messages content in a txt file.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file has_simple_printer.h.

13.242 has_simple_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file has_simple_printer.h
00003  * \brief Interface of a class that prints HAS messages content in a txt file.
00004  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_HAS_SIMPLE_PRINTER_H
00018 #define GNSS_SDR_HAS_SIMPLE_PRINTER_H
00019
00020 #include <cstddef>  // for size_t
00021 #include <fstream>  // for std::ofstream
00022 #include <mutex>    // for std::mutex
00023 #include <string>   // for std::string
00024 #include <vector>   // for std::vector
00025
00026 /** \addtogroup PVT
00027  * \{ */
00028 /** \addtogroup PVT_libs
00029  * \{ */
00030
00031 class Galileo_HAS_data;
00032
00033 /*!
00034  * \brief Prints HAS messages content in a txt file. See HAS-SIS-ICD for a
00035  * message description.
00036  */
00037 class Has_Simple_Printer
00038 {
00039 public:
00040     explicit Has_Simple_Printer(const std::string& base_path = std::string("."), const std::string&
      filename = std::string("HAS_Messages"), bool time_tag_name = true);
00041     ~Has_Simple_Printer() noexcept;
00042     bool print_message(const Galileo_HAS_data* const has_data);
00043
00044 private:
00045     template <class T>
00046     std::string print_vector(const std::vector<T>& vec, float scale_factor = 1) const;
00047
00048     template <class T>
00049     std::string print_vector_binary(const std::vector<T>& vec, size_t bit_length) const;
00050
00051     template <class T>
00052     std::string print_matrix(const std::vector<std::vector<T>& mat, const std::string& filler, float
      scale_factor = 1) const;
00053
00054     std::string print_vector_string(const std::vector<std::string>& vec) const;
00055
00056     bool close_file();
00057
00058     std::mutex d_mutex;
00059     std::ofstream d_has_file;
00060     std::string d_has_filename;
00061     std::string d_has_base_path;
00062     bool d_data_printed;
00063 };
00064
00065 /** \} */
00066 /** \} */
00067 #endif  // GNSS_SDR_HAS_SIMPLE_PRINTER_H
```

13.243 kml_printer.h File Reference

Interface of a class that prints PVT information to a kml file.

```
#include <fstream>
#include <string>
```

Classes

- class Kml_Printer

  Prints PVT information to OGC KML format file (can be viewed with Google Earth).

13.243.1 Detailed Description

Interface of a class that prints PVT information to a kml file.

Author

Javier Arribas, 2011. jarribas(at)cttc.es Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file kml_printer.h.

13.244 kml_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file kml_printer.h
00003  * \brief Interface of a class that prints PVT information to a kml file
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *         Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_KML_PRINTER_H
00020 #define GNSS_SDR_KML_PRINTER_H
00021
00022 #include <fstream>  // for ofstream
00023 #include <string>
00024
00025 /** \addtogroup PVT
00026  * \{ */
00027 /** \addtogroup PVT_libs
00028  * \{ */
00029
00030
00031 class Pvt_Solution;
00032
00033 /*!
00034  * \brief Prints PVT information to OGC KML format file (can be viewed with Google Earth)
00035  *
00036  * See https://www.opengeospatial.org/standards/kml
00037  */
00038 class Kml_Printer
00039 {
00040 public:
00041     explicit Kml_Printer(const std::string& base_path = std::string("."));
00042     ~Kml_Printer() noexcept;
00043     bool set_headers(const std::string& filename, bool time_tag_name = true);
00044     bool print_position(const Pvt_Solution* const position);
00045     bool close_file();
00046
00047 private:
00048     std::ofstream kml_file;
```

```
00049     std::ofstream tmp_file;
00050     std::string kml_filename;
00051     std::string kml_base_path;
00052     std::string tmp_file_str;
00053     std::string indent;
00054     unsigned int point_id;
00055     bool positions_printed;
00056 };
00057
00058
00059 /** \} */
00060 /** \} */
00061 #endif  // GNSS_SDR_KML_PRINTER_H
```

13.245 monitor_ephemeris_udp_sink.h File Reference

Interface of a class that sends serialized Gps_Ephemeris and Galileo_Ephemeris objects over udp to one or multiple endpoints.

```
#include "galileo_ephemeris.h"
#include "gps_ephemeris.h"
#include "serdes_galileo_eph.h"
#include "serdes_gps_eph.h"
#include <boost/asio.hpp>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class Monitor_Ephemeris_Udp_Sink

Typedefs

- using b_io_context = boost::asio::io_service

13.245.1 Detailed Description

Interface of a class that sends serialized Gps_Ephemeris and Galileo_Ephemeris objects over udp to one or multiple endpoints.

Author

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file monitor_ephemeris_udp_sink.h.

13.246 monitor_ephemeris_udp_sink.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file monitor_ephemeris_udp_sink.h
00003  * \brief Interface of a class that sends serialized Gps_Ephemeris and
00004  * Galileo_Ephemeris objects over udp to one or multiple endpoints.
00005  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
```

```
00017
00018 #ifndef GNSS_SDR_MONITOR_EPHEMERIS_UDP_SINK_H
00019 #define GNSS_SDR_MONITOR_EPHEMERIS_UDP_SINK_H
00020
00021 #include "galileo_ephemeris.h"
00022 #include "gps_ephemeris.h"
00023 #include "serdes_galileo_eph.h"
00024 #include "serdes_gps_eph.h"
00025 #include <boost/asio.hpp>
00026 #include <memory>
00027 #include <string>
00028 #include <vector>
00029
00030 /** \addtogroup PVT
00031  * \{ */
00032 /** \addtogroup PVT_libs
00033  * \{ */
00034
00035
00036 #if USE_BOOST_ASIO_IO_CONTEXT
00037 using b_io_context = boost::asio::io_context;
00038 #else
00039 using b_io_context = boost::asio::io_service;
00040 #endif
00041
00042 class Monitor_Ephemeris_Udp_Sink
00043 {
00044 public:
00045     Monitor_Ephemeris_Udp_Sink(const std::vector<std::string>& addresses, const uint16_t& port, bool
     protobuf_enabled);
00046     bool write_gps_ephemeris(const std::shared_ptr<Gps_Ephemeris>& monitor_gps_eph);
00047     bool write_galileo_ephemeris(const std::shared_ptr<Galileo_Ephemeris>& monitor_gal_eph);
00048
00049 private:
00050     Serdes_Galileo_Eph serdes_gal;
00051     Serdes_Gps_Eph serdes_gps;
00052     b_io_context io_context;
00053     boost::asio::ip::udp::socket socket;
00054     std::vector<boost::asio::ip::udp::endpoint> endpoints;
00055     boost::system::error_code error;
00056     bool use_protobuf;
00057 };
00058
00059
00060 /** \} */
00061 /** \} */
00062 #endif  // GNSS_SDR_MONITOR_EPHEMERIS_UDP_SINK_H
```

13.247 monitor_pvt.h File Reference

Interface of the Monitor_Pvt class.

```
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/vector.hpp>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class Monitor_Pvt

  This class contains parameters and outputs of the PVT block.

- class Monitor_Pvt::TrackedSatelliteInfo

  One tracked satellite/signal with its azimuth/elevation, whether it was combined with another signal of the same satellite (e.g. Galileo E1+E5a iono-free combination) and whether it was used in this fix. Signals are listed individually (one entry per satellite per signal): a combined satellite appears as two entries, both with combined = true.

13.247.1 Detailed Description

Interface of the Monitor_Pvt class.

Author

13.247.1.1 Álvaro Cebrián Juan, 2019. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file monitor_pvt.h.

13.248 monitor_pvt.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file monitor_pvt.h
00003  * \brief  Interface of the Monitor_Pvt class
00004  * \author
00005  *  Álvaro Cebrián Juan, 2019. acebrianjuan(at)gmail.com
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_MONITOR_PVT_H
00018 #define GNSS_SDR_MONITOR_PVT_H
00019
00020 #include <boost/serialization/nvp.hpp>
00021 #include <boost/serialization/vector.hpp>
00022 #include <cstdint>
00023 #include <string>
00024 #include <vector>
00025
00026 /** \addtogroup PVT
00027  * \{ */
00028 /** \addtogroup PVT_libs
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class contains parameters and outputs of the PVT block
00034  */
00035 class Monitor_Pvt
00036 {
00037 public:
00038     /*!
00039      * \brief One tracked satellite/signal with its azimuth/elevation, whether
00040      * it was combined with another signal of the same satellite (e.g. Galileo
00041      * E1+E5a iono-free combination) and whether it was used in this fix.
00042      * Signals are listed individually (one entry per satellite per signal):
00043      * a combined satellite appears as two entries, both with combined = true.
00044      */
00045     class TrackedSatelliteInfo
00046     {
00047     public:
00048         uint32_t prn{};
00049         char system{};  // 'G' GPS, 'E' Galileo, 'R' GLONASS, 'C' BeiDou, 'S' SBAS, 'J' QZSS
00050         std::string signal;
00051         double azimuth_deg{};
00052         double elevation_deg{};
00053         bool combined{};
00054         // false when tracked (az/el valid) but excluded from the solve, e.g.
00055         // below PVT.elevation_mask or by RAIM FDE.
00056         bool used{true};
00057         // Broadcast health status, independent of `used` (which alone can't
00058         // tell a monitor client *why* a satellite was excluded).
00059         bool healthy{true};
00060
00061         template <class Archive>
00062         void serialize(Archive& ar, const unsigned int version)
00063         {
00064             if (version)
00065                 {
00066                 };
00067             ar& BOOST_SERIALIZATION_NVP(prn);
00068             ar& BOOST_SERIALIZATION_NVP(system);
```

```
00069             ar& BOOST_SERIALIZATION_NVP(signal);
00070             ar& BOOST_SERIALIZATION_NVP(azimuth_deg);
00071             ar& BOOST_SERIALIZATION_NVP(elevation_deg);
00072             ar& BOOST_SERIALIZATION_NVP(combined);
00073             ar& BOOST_SERIALIZATION_NVP(used);
00074             ar& BOOST_SERIALIZATION_NVP(healthy);
00075         }
00076     };
00077
00078     std::vector<TrackedSatelliteInfo> tracked_satellites;
00079
00080     // TOW
00081     uint32_t TOW_at_current_symbol_ms;
00082     // WEEK
00083     uint32_t week;
00084     // PVT GPS time
00085     double RX_time;
00086     // User clock offset [s]
00087     double user_clk_offset;
00088
00089     // ECEF POS X,Y,X [m] + ECEF VEL X,Y,X [m/s] (6 x double)
00090     double pos_x;
00091     double pos_y;
00092     double pos_z;
00093     double vel_x;
00094     double vel_y;
00095     double vel_z;
00096
00097     // position variance/covariance (m^2) {c_xx,c_yy,c_zz,c_xy,c_yz,c_zx} (6 x double)
00098     double cov_xx;
00099     double cov_yy;
00100     double cov_zz;
00101     double cov_xy;
00102     double cov_yz;
00103     double cov_zx;
00104
00105     // GEO user position Latitude [deg]
00106     double latitude;
00107     // GEO user position Longitude [deg]
00108     double longitude;
00109     // GEO user position Height [m]
00110     double height;
00111     // East, Nord, Up (ENU) Velocity [m/s]
00112     double vel_e;
00113     double vel_n;
00114     double vel_u;
00115
00116     // Course Over Ground (COG) [deg]
00117     double cog;
00118
00119     // Galileo HAS status: 1- HAS messages decoded and applied, 0 - HAS not available
00120     uint32_t galhas_status;
00121
00122     // NUMBER OF VALID SATS
00123     uint8_t valid_sats;
00124     // RTKLIB solution status
00125     uint8_t solution_status;
00126     // RTKLIB solution type (0:xyz-ecef,1:enu-baseline)
00127     uint8_t solution_type;
00128     // AR ratio factor for validation
00129     float AR_ratio_factor;
00130     // AR ratio threshold for validation
00131     float AR_ratio_threshold;
00132
00133     // GDOP / PDOP/ HDOP/ VDOP
00134     double gdop;
00135     double pdop;
00136     double hdop;
00137     double vdop;
00138
00139     // User clock drift [ppm]
00140     double user_clk_drift_ppm;
00141
00142     // PVT UTC Time (rfc 3339 datetime string)
00143     std::string utc_time;
00144
00145     std::string geohash;  // See https://en.wikipedia.org/wiki/Geohash
00146
00147     /*!
00148      * \brief This member function serializes and restores
00149      * Monitor_Pvt objects from a byte stream.
00150      */
00151     template <class Archive>
00152
00153     void serialize(Archive& ar, const unsigned int version)
00154     {
00155         if (version)
```

```
00156             {
00157             };
00158
00159         ar& BOOST_SERIALIZATION_NVP(TOW_at_current_symbol_ms);
00160         ar& BOOST_SERIALIZATION_NVP(week);
00161         ar& BOOST_SERIALIZATION_NVP(RX_time);
00162         ar& BOOST_SERIALIZATION_NVP(user_clk_offset);
00163
00164         ar& BOOST_SERIALIZATION_NVP(pos_x);
00165         ar& BOOST_SERIALIZATION_NVP(pos_y);
00166         ar& BOOST_SERIALIZATION_NVP(pos_z);
00167         ar& BOOST_SERIALIZATION_NVP(vel_x);
00168         ar& BOOST_SERIALIZATION_NVP(vel_y);
00169         ar& BOOST_SERIALIZATION_NVP(vel_z);
00170
00171         ar& BOOST_SERIALIZATION_NVP(cov_xx);
00172         ar& BOOST_SERIALIZATION_NVP(cov_yy);
00173         ar& BOOST_SERIALIZATION_NVP(cov_zz);
00174         ar& BOOST_SERIALIZATION_NVP(cov_xy);
00175         ar& BOOST_SERIALIZATION_NVP(cov_yz);
00176         ar& BOOST_SERIALIZATION_NVP(cov_zx);
00177
00178         ar& BOOST_SERIALIZATION_NVP(latitude);
00179         ar& BOOST_SERIALIZATION_NVP(longitude);
00180         ar& BOOST_SERIALIZATION_NVP(height);
00181
00182         ar& BOOST_SERIALIZATION_NVP(valid_sats);
00183         ar& BOOST_SERIALIZATION_NVP(solution_status);
00184         ar& BOOST_SERIALIZATION_NVP(solution_type);
00185         ar& BOOST_SERIALIZATION_NVP(AR_ratio_factor);
00186         ar& BOOST_SERIALIZATION_NVP(AR_ratio_threshold);
00187
00188         ar& BOOST_SERIALIZATION_NVP(gdop);
00189         ar& BOOST_SERIALIZATION_NVP(pdop);
00190         ar& BOOST_SERIALIZATION_NVP(hdop);
00191         ar& BOOST_SERIALIZATION_NVP(vdop);
00192
00193         ar& BOOST_SERIALIZATION_NVP(user_clk_drift_ppm);
00194         ar& BOOST_SERIALIZATION_NVP(utc_time);
00195
00196         ar& BOOST_SERIALIZATION_NVP(vel_e);
00197         ar& BOOST_SERIALIZATION_NVP(vel_n);
00198         ar& BOOST_SERIALIZATION_NVP(vel_u);
00199
00200         ar& BOOST_SERIALIZATION_NVP(cog);
00201         ar& BOOST_SERIALIZATION_NVP(geohash);
00202         ar& BOOST_SERIALIZATION_NVP(tracked_satellites);
00203     }
00204 };
00205
00206
00207 /** \} */
00208 /** \} */
00209 #endif  // GNSS_SDR_MONITOR_PVT_H
```

13.249 monitor_pvt_udp_sink.h File Reference

Interface of a class that sends serialized Monitor_Pvt objects over udp to one or multiple endpoints.

```
#include "monitor_pvt.h"
#include "serdes_monitor_pvt.h"
#include <boost/asio.hpp>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class Monitor_Pvt_Udp_Sink

Typedefs

- using b_io_context = boost::asio::io_service

13.249.1 Detailed Description

Interface of a class that sends serialized Monitor_Pvt objects over udp to one or multiple endpoints.

Author

Álvaro Cebrián Juan, 2019. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file monitor_pvt_udp_sink.h.

13.250 monitor_pvt_udp_sink.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file monitor_pvt_udp_sink.h
00003  * \brief Interface of a class that sends serialized Monitor_Pvt objects
00004  * over udp to one or multiple endpoints
00005  * \author Álvaro Cebrián Juan, 2019. acebrianjuan(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_MONITOR_PVT_UDP_SINK_H
00019 #define GNSS_SDR_MONITOR_PVT_UDP_SINK_H
00020
00021 #include "monitor_pvt.h"
00022 #include "serdes_monitor_pvt.h"
00023 #include <boost/asio.hpp>
00024 #include <memory>
00025 #include <string>
00026 #include <vector>
00027
00028 /** \addtogroup PVT
00029  * \{ */
00030 /** \addtogroup PVT_libs
00031  * \{ */
00032
00033
00034 #if USE_BOOST_ASIO_IO_CONTEXT
00035 using b_io_context = boost::asio::io_context;
00036 #else
00037 using b_io_context = boost::asio::io_service;
00038 #endif
00039
00040 class Monitor_Pvt_Udp_Sink
00041 {
00042 public:
00043     Monitor_Pvt_Udp_Sink(
00044         const std::vector<std::string>& addresses,
00045         const std::vector<std::string>& ports,
00046         bool protobuf_enabled);
00047     bool write_monitor_pvt(const Monitor_Pvt* const monitor_pvt);
00048
00049 private:
00050     Serdes_Monitor_Pvt serdes;
00051     b_io_context io_context;
00052     boost::asio::ip::udp::socket socket;
00053     std::vector<boost::asio::ip::udp::endpoint> endpoints;
00054     boost::system::error_code error;
00055     bool use_protobuf;
00056 };
00057
00058
00059 /** \} */
00060 /** \} */
00061 #endif  // GNSS_SDR_MONITOR_PVT_UDP_SINK_H
```

13.251 nmea_printer.h File Reference

Interface of a NMEA 2.1 printer for GNSS-SDR This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA). See https://www.nmea.org/ for the NMEA 183 standard.

```
#include <boost/date_time/posix_time/ptime.hpp>
#include <fstream>
#include <memory>
#include <string>
```

Classes

- class Nmea_Printer

  This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA).

13.251.1 Detailed Description

Interface of a NMEA 2.1 printer for GNSS-SDR This class provides a implementation of a subset of the NMEA-0183 standard for interfacing marine electronic devices as defined by the National Marine Electronics Association (NMEA). See https://www.nmea.org/ for the NMEA 183 standard.

Author

Javier Arribas, 2012. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file nmea_printer.h.

13.252 nmea_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nmea_printer.h
00003  * \brief Interface of a NMEA 2.1 printer for GNSS-SDR
00004  * This class provides a implementation of a subset of the NMEA-0183 standard for interfacing
00005  * marine electronic devices as defined by the National Marine Electronics Association (NMEA).
00006  * See https://www.nmea.org/ for the NMEA 183 standard
00007  *
00008  * \author Javier Arribas, 2012. jarribas(at)cttc.es
00009  *
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_NMEA_PRINTER_H
00023 #define GNSS_SDR_NMEA_PRINTER_H
00024
00025 #include <boost/date_time/posix_time/ptime.hpp>  // for ptime
00026 #include <fstream>                               // for ofstream
00027 #include <memory>                                // for shared_ptr
00028 #include <string>                                // for string
00029
00030 /** \addtogroup PVT
00031  * \{ */
00032 /** \addtogroup PVT_libs
00033  * \{ */
00034
00035
00036 class Rtklib_Solver;
00037
00038 /*!
```

```
00039  * \brief This class provides a implementation of a subset of the NMEA-0183 standard for interfacing
00040  * marine electronic devices as defined by the National Marine Electronics Association (NMEA).
00041  *
00042  * See https://en.wikipedia.org/wiki/NMEA_0183
00043  */
00044 class Nmea_Printer
00045 {
00046 public:
00047     /*!
00048      * \brief Default constructor.
00049      */
00050     Nmea_Printer(const std::string& filename, bool flag_nmea_output_file, bool flag_nmea_tty_port,
      std::string nmea_dump_devname, const std::string& base_path = ".");
00051
00052     /*!
00053      * \brief Default destructor.
00054      */
00055     ~Nmea_Printer() noexcept;
00056
00057     /*!
00058      * \brief Print NMEA PVT and satellite info to the initialized device
00059      */
00060     bool Print_Nmea_Line(const Rtklib_Solver* const pvt_data);
00061
00062     /*!
00063      * \brief Returns GPGGA message (fix data)
00064      */
00065     std::string get_GPGGA(const Rtklib_Solver* const pvt_data) const;
00066
00067     /*!
00068      * \brief Returns GPGSA message (overall satellite reception data)
00069      */
00070     std::string get_GPGSA(const Rtklib_Solver* const pvt_data) const;
00071
00072 private:
00073     int init_serial(const std::string& serial_device);  // serial port control
00074     void close_serial() const;
00075     std::string get_GPGGA() const;  // fix data
00076     std::string get_GPGSV() const;  // satellite data
00077     std::string get_GPGSA() const;  // overall satellite reception data
00078     std::string get_GPRMC() const;  // minimum recommended data
00079     std::string get_UTC_NMEA_time(const boost::posix_time::ptime d_position_UTC_time) const;
00080     std::string longitude_to_hm(double longitude) const;
00081     std::string latitude_to_hm(double lat) const;
00082     char checkSum(const std::string& sentence) const;
00083
00084     const Rtklib_Solver* d_PVT_data;
00085
00086     std::ofstream nmea_file_descriptor;  // Output file stream for NMEA log file
00087
00088     std::string nmea_filename;  // String with the NMEA log filename
00089     std::string nmea_base_path;
00090     std::string nmea_devname;
00091
00092     int nmea_dev_descriptor;  // NMEA serial device descriptor (i.e. COM port)
00093     bool d_flag_nmea_output_file;
00094 };
00095
00096
00097 /** \} */
00098 /** \} */
00099 #endif  // GNSS_SDR_NMEA_PRINTER_H
```

13.253 ntrip_rtcm_client.h File Reference

NTRIP client for RTCM3 fixed-base corrections.

```
#include "pvt_conf.h"
#include "rtklib.h"
#include <array>
#include <cstddef>
#include <cstdint>
#include <functional>
#include <memory>
#include <string>
#include <vector>
```

Classes

- struct Ntrip_Rtcm_Client_Config
- struct Ntrip_Rtcm_Client_Status
- struct Ntrip_Rtcm_Snapshot
- class Ntrip_Rtcm_Client

  Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster, preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced v1, optionally over TLS.

Enumerations

- enum class **Ntrip_Rtcm_Client_State** {
  DISABLED , **STOPPED** , **STARTING** , **CONNECTING** ,
  AUTHENTICATING , **STREAMING** , **RECONNECT_WAIT** , **ERROR** ,
  STOPPING }

Functions

- void secure_clear_ntrip_credentials (Pvt_Conf ∗conf) noexcept

  Scrubs the NTRIP credential fields of a Pvt_Conf copy ahead of its destruction (the Secure_String fields wipe themselves on release, but a long-lived copy should drop its secrets as soon as they are consumed). Single home of the credential field list.
- bool ntrip_text_has_space_or_control (const std::string &text)

  True when the text contains whitespace or control characters — the character rule shared by the credential validation and the adapter's environment-variable-name check.
- Ntrip_Rtcm_Client_Config make_ntrip_rtcm_client_config (const Pvt_Conf &conf)

  Builds the NTRIP client configuration from the PVT configuration — the single home of the field mapping, shared by the adapter's validation and by the PVT block's client construction. The mountpoint is stored normalized; validate() and the transport-path builder normalize again defensively for directly-built configurations.

13.253.1 Detailed Description

NTRIP client for RTCM3 fixed-base corrections.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ntrip_rtcm_client.h.

13.254 ntrip_rtcm_client.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ntrip_rtcm_client.h
00003  * \brief NTRIP client for RTCM3 fixed-base corrections
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_NTRIP_RTCM_CLIENT_H
00018 #define GNSS_SDR_NTRIP_RTCM_CLIENT_H
00019
00020 #include "pvt_conf.h"
```

```
00021 #include "rtklib.h"
00022 #include <array>
00023 #include <cstddef>
00024 #include <cstdint>
00025 #include <functional>
00026 #include <memory>
00027 #include <string>
00028 #include <vector>
00029
00030 class Ntrip_Rtcm_Client_Test_Access;
00031
00032 /** \addtogroup PVT
00033  * \{ */
00034 /** \addtogroup PVT_libs
00035  * \{ */
00036
00037 //! Scrubs the NTRIP credential fields of a Pvt_Conf copy ahead of its
00038 //! destruction (the Secure_String fields wipe themselves on release, but a
00039 //! long-lived copy should drop its secrets as soon as they are consumed).
00040 //! Single home of the credential field list.
00041 void secure_clear_ntrip_credentials(Pvt_Conf* conf) noexcept;
00042
00043 //! True when the text contains whitespace or control characters -- the
00044 //! character rule shared by the credential validation and the adapter's
00045 //! environment-variable-name check.
00046 bool ntrip_text_has_space_or_control(const std::string& text);
00047
00048 struct Ntrip_Rtcm_Client_Config
00049 {
00050     bool enabled = false;
00051     std::string host;
00052     std::uint16_t port = 2101;
00053     std::string mountpoint;
00054     // self-wiping: every copy of this configuration scrubs its credentials on
00055     // destruction, so holders need no manual scrub site
00056     Secure_String username;
00057     Secure_String password;
00058     // Zero disables operational reconnects, but not the single immediate v2-to-v1
00059     // compatibility retry; positive values select the reconnect delay.
00060     int reconnect_interval_ms = 10000;
00061     int timeout_ms = 10000;
00062     double max_age_s = 5.0;
00063     bool send_gga = false;
00064     int gga_period_ms = 10000;
00065
00066     // NTRIP protocol version requested from the transport: 2 prefers v2,
00067     // accepts a legacy reply, and retries once on a fresh v1 connection after
00068     // HTTP 400/501/505 or no response bytes; 1 forces legacy v1.
00069     int version = 2;
00070
00071     // Enables TLS 1.2-or-newer transport with certificate validation against
00072     // the system CA store and host name verification.
00073     bool tls_enabled = false;
00074
00075     // Zero accepts any stream station while keeping observations and position
00076     // bound to one ID. A positive value filters to that exact 12-bit ID.
00077     int station_id = 0;
00078
00079     //! Returns an empty string when the configuration is usable, or the
00080     //! reason it is not. Single source of the NTRIP configuration rules,
00081     //! shared by the adapter (which turns the reason into a throw at
00082     //! constructor time) and by start() (which turns it into an ERROR state).
00083     std::string validate() const;
00084 };
00085
00086
00087 //! Builds the NTRIP client configuration from the PVT configuration -- the
00088 //! single home of the field mapping, shared by the adapter's validation and
00089 //! by the PVT block's client construction. The mountpoint is stored
00090 //! normalized; validate() and the transport-path builder normalize again
00091 //! defensively for directly-built configurations.
00092 Ntrip_Rtcm_Client_Config make_ntrip_rtcm_client_config(const Pvt_Conf& conf);
00093
00094
00095 enum class Ntrip_Rtcm_Client_State
00096 {
00097     DISABLED,
00098     STOPPED,
00099     STARTING,
00100     CONNECTING,
00101     AUTHENTICATING,
00102     STREAMING,
00103     RECONNECT_WAIT,
00104     ERROR,
00105     STOPPING
00106 };
00107
```

```
00108
00109 struct Ntrip_Rtcm_Client_Status
00110 {
00111     Ntrip_Rtcm_Client_State state = Ntrip_Rtcm_Client_State::STOPPED;
00112     bool running = false;
00113     bool connected = false;
00114     std::string message;
00115     std::uint64_t bytes_received = 0;
00116     std::uint64_t rtcm_messages_decoded = 0;
00117     std::uint64_t observation_epochs_decoded = 0;
00118     std::uint64_t decoder_errors = 0;
00119     std::uint64_t reconnect_count = 0;
00120 };
00121
00122
00123 struct Ntrip_Rtcm_Snapshot
00124 {
00125     bool has_observations = false;
00126     std::vector<obsd_t> observations;
00127     gtime_t observation_time = {0, 0.0};
00128
00129     // Signed reference-minus-base time difference. Freshness uses its absolute
00130     // value, so a small future-dated base epoch is accepted as well.
00131     double age_s = 0.0;
00132     bool fresh = false;
00133     // Bumped on every output-side change (observations, base position,
00134     // ephemerides, reset), so equal values mean an identical snapshot.
00135     std::uint64_t generation = 0;
00136
00137     bool has_base_position = false;
00138     std::array<double, 3> base_position_ecef_m = {{0.0, 0.0, 0.0}};
00139     int base_position_message_type = 0;
00140     int station_id = 0;
00141     int itrf = 0;
00142     double antenna_height_m = 0.0;
00143
00144     // Latest decoded broadcast ephemeris for each GPS, Galileo or BeiDou
00145     // satellite seen on the correction stream (RTCM MT1019/1045/1046/1042),
00146     // sorted by satellite number for binary lookup. The records own no
00147     // dynamic memory.
00148     std::vector<eph_t> ephemerides;
00149
00150     //! Recomputes the rover-time-dependent fields age_s and fresh. A cached
00151     //! copy must call this per epoch: the generation covers the received
00152     //! data, not the passage of rover time.
00153     void refresh_age(const gtime_t& rover_time, double max_age_s);
00154 };
00155
00156
00157 /*!
00158  * \brief Receives and decodes a fixed-base RTCM3 stream from an NTRIP caster,
00159  * preferring NTRIP v2 with a fresh-connection fallback to v1, or using forced
00160  * v1, optionally over TLS.
00161  *
00162  * The object owns one worker thread. The public snapshot and status methods
00163  * copy data under a mutex and can be called concurrently with the worker.
00164  */
00165 class Ntrip_Rtcm_Client
00166 {
00167 public:
00168     explicit Ntrip_Rtcm_Client(const Ntrip_Rtcm_Client_Config& config);
00169     ~Ntrip_Rtcm_Client() noexcept;
00170
00171     Ntrip_Rtcm_Client(const Ntrip_Rtcm_Client&) = delete;
00172     Ntrip_Rtcm_Client& operator=(const Ntrip_Rtcm_Client&) = delete;
00173     Ntrip_Rtcm_Client(Ntrip_Rtcm_Client&&) = delete;
00174     Ntrip_Rtcm_Client& operator=(Ntrip_Rtcm_Client&&) = delete;
00175
00176     // Returns false when disabled, invalidly configured, or if the worker
00177     // thread could not be created. Calling start() on a running client is safe.
00178     bool start();
00179     void stop() noexcept;
00180     bool running() const noexcept;
00181     Ntrip_Rtcm_Client_Status status() const;
00182
00183     // The no-argument overload evaluates age against current GPST. The rover
00184     // overload should be used by an RTK solver so recorded data does not depend
00185     // on wall-clock time.
00186     Ntrip_Rtcm_Snapshot latest_snapshot() const;
00187     Ntrip_Rtcm_Snapshot latest_snapshot(const gtime_t& rover_time) const;
00188
00189     // Cheap change detectors so per-epoch callers can skip the deep copies:
00190     // snapshot_generation() matches Ntrip_Rtcm_Snapshot::generation, and
00191     // state_generation() changes whenever the status state/message is set.
00192     std::uint64_t snapshot_generation() const noexcept;
00193     std::uint64_t state_generation() const noexcept;
00194
```

```
00195     // Position is ECEF in metres. Invalid/non-finite positions disable GGA
00196     // until another valid position is supplied.
00197     void update_rover_position(const std::array<double, 3>& position_ecef_m);
00198     void update_rover_position(const std::array<double, 3>& position_ecef_m,
00199         const gtime_t& rover_time);
00200
00201     // Supplies the approximate GPST used to resolve RTCM epoch rollovers. It
00202     // can be called before start() and updated once per rover observation epoch.
00203     void update_rover_time(const gtime_t& rover_time);
00204
00205 private:
00206     friend class Ntrip_Rtcm_Client_Test_Access;
00207
00208     using Hostname_Resolver =
00209         std::function<bool(const std::string&, std::string*)>;
00210
00211     // Test-only resolver injection. Production callers always use the system
00212     // IPv4 resolver installed by the implementation constructor.
00213     bool set_hostname_resolver_for_test(Hostname_Resolver resolver);
00214     // Synchronized process-wide observations used by resolver lifecycle tests.
00215     static std::uint64_t hostname_resolution_launch_count_for_test();
00216     static std::size_t in_flight_hostname_resolution_count_for_test();
00217
00218     class Impl;
00219     std::unique_ptr<Impl> d_impl;
00220 };
00221
00222 /** \} */
00223 /** \} */
00224
00225 #endif  // GNSS_SDR_NTRIP_RTCM_CLIENT_H
```

13.255 ntrip_rtk_signals.h File Reference

Signal table of the NTRIP RTK path.

```
#include "rtklib.h"
#include "signal_enabled_flags.h"
#include "signal_flag.h"
#include <array>
#include <cstdint>
#include <string>
```

Classes

- struct Ntrip_Rtk_Signal

Functions

- bool ntrip_rtk_is_entry_band (const Ntrip_Rtk_Signal &signal)

  The entry band of every constellation occupies RTKLIB's first slot, so "first band" is band_slot == 0 by definition — a fact, not a column that could drift from the slot assignment.
- const std::array< Ntrip_Rtk_Signal, 6 > & ntrip_rtk_signals ()
- uint32_t ntrip_rtk_supported_signal_mask ()

  Bitmask of every signal_flag the NTRIP RTK path supports.
- int ntrip_rtk_navigation_systems (const Signal_Enabled_Flags &enabled)

  RTKLIB navigation-system mask spanned by the enabled NTRIP RTK channels.
- int ntrip_rtk_required_bands (const Signal_Enabled_Flags &enabled)

  Number of RTKLIB frequency slots the enabled NTRIP RTK channels require.
- int ntrip_rtk_second_band_slot (int system, const Signal_Enabled_Flags &enabled)
- const char * ntrip_rtk_system_label (int system)

  Constellation label of an RTKLIB system bit, for messages.
- bool ntrip_rtk_valid_channel_set (const Signal_Enabled_Flags &enabled)
- std::string ntrip_rtk_supported_sets_description ()

  Human-readable list of the supported per-system channel sets, derived from the table for the adapter's rejection message.

13.255.1 Detailed Description

Signal table of the NTRIP RTK path.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: BSD-2-Clause
Definition in file ntrip_rtk_signals.h.

13.256 ntrip_rtk_signals.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ntrip_rtk_signals.h
00003  * \brief Signal table of the NTRIP RTK path
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: BSD-2-Clause
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_NTRIP_RTK_SIGNALS_H
00018 #define GNSS_SDR_NTRIP_RTK_SIGNALS_H
00019
00020 #include "rtklib.h"
00021 #include "signal_enabled_flags.h"
00022 #include "signal_flag.h"
00023 #include <array>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup PVT
00028  * \{ */
00029 /** \addtogroup PVT_libs
00030  * \{ */
00031
00032 /*!
00033  * Single home for the signal knowledge of the NTRIP RTK path: which tracked
00034  * signals it supports and, for each one, the constellation, the RTKLIB
00035  * frequency slot it occupies, and the channel signal code. The adapter's
00036  * channel-set validity predicate, its error message, its derived slot count
00037  * and navigation-system mask, and the client's per-slot observation filter
00038  * all derive from this table, and the solver cross-checks its own band
00039  * mapping against it at construction -- so admitting a new SIGNAL of a
00040  * supported constellation is a one-row change (plus documentation and
00041  * tests; a slot-sharing preference rule like BeiDou's prefer-B1C stays an
00042  * explicit exception in the client). A new SYSTEM additionally needs
00043  * base-side admission in the client and the solver, and navigation-system
00044  * compatibility handling in the adapter.
00045  */
00046 struct Ntrip_Rtk_Signal
00047 {
00048     uint32_t flag;       //!< signal_flag bit of the tracked channel signal
00049     int system;          //!< RTKLIB constellation bit (SYS_XXX)
00050     int band_slot;       //!< RTKLIB frequency slot the signal occupies (0-based)
00051     const char* signal;  //!< two-character channel signal code (Gnss_Synchro convention)
00052 };
00053
00054
00055 //! The entry band of every constellation occupies RTKLIB's first slot, so
00056 //! "first band" is band_slot == 0 by definition -- a fact, not a column that
00057 //! could drift from the slot assignment
00058 inline bool ntrip_rtk_is_entry_band(const Ntrip_Rtk_Signal& signal)
00059 {
00060     return signal.band_slot == 0;
00061 }
00062
00063
00064 /*!
00065  * The signal table. A function-local static keeps the header buildable as
00066  * C++11 (an inline variable would need C++17) while still providing a single
00067  * instance. Keep each constellation's second bands ordered by preference:
```

```
00068  * with more than one enabled (a configuration the adapter rejects today),
00069  * ntrip_rtk_second_band_slot() picks the first match.
00070  */
00071 inline const std::array<Ntrip_Rtk_Signal, 6>& ntrip_rtk_signals()
00072 {
00073     static const std::array<Ntrip_Rtk_Signal, 6> table = {{
00074         {GPS_1C, SYS_GPS, 0, "1C"},   // GPS L1 C/A
00075         {GPS_2S, SYS_GPS, 1, "2S"},   // GPS L2C
00076         {GPS_L5, SYS_GPS, 2, "L5"},   // GPS L5 (shares the third slot with E5a)
00077         {GAL_1B, SYS_GAL, 0, "1B"},   // Galileo E1
00078         {GAL_E5a, SYS_GAL, 2, "5X"},  // Galileo E5a
00079         {BDS_B1C, SYS_BDS, 0, "1D"},  // BeiDou B1C
00080     }};
00081     return table;
00082 }
00083
00084
00085 //! Bitmask of every signal_flag the NTRIP RTK path supports
00086 inline uint32_t ntrip_rtk_supported_signal_mask()
00087 {
00088     uint32_t mask = 0;
00089     for (const auto& signal : ntrip_rtk_signals())
00090         {
00091             mask |= signal.flag;
00092         }
00093     return mask;
00094 }
00095
00096
00097 //! RTKLIB navigation-system mask spanned by the enabled NTRIP RTK channels
00098 inline int ntrip_rtk_navigation_systems(const Signal_Enabled_Flags& enabled)
00099 {
00100     int navsys = 0;
00101     for (const auto& signal : ntrip_rtk_signals())
00102         {
00103             if (ntrip_rtk_is_entry_band(signal) && enabled.check_any_enabled(signal.flag))
00104                 {
00105                     navsys |= signal.system;
00106                 }
00107         }
00108     return navsys;
00109 }
00110
00111
00112 //! Number of RTKLIB frequency slots the enabled NTRIP RTK channels require
00113 inline int ntrip_rtk_required_bands(const Signal_Enabled_Flags& enabled)
00114 {
00115     int bands = 1;
00116     for (const auto& signal : ntrip_rtk_signals())
00117         {
00118             if (enabled.check_any_enabled(signal.flag) && signal.band_slot + 1 > bands)
00119                 {
00120                     bands = signal.band_slot + 1;
00121                 }
00122         }
00123     return bands;
00124 }
00125
00126
00127 /*!
00128  * RTKLIB slot of a satellite system's second band given the enabled channels.
00129  * QZSS rides the GPS bands. When no second band of the system is enabled, the
00130  * constellation's canonical second slot is returned (Galileo/BeiDou: 2, both
00131  * anticipate slot-2 signals; GPS/QZSS: 1) so unsupported slots can be cleared.
00132  */
00133 inline int ntrip_rtk_second_band_slot(int system, const Signal_Enabled_Flags& enabled)
00134 {
00135     const int lookup_system = (system == SYS_QZS) ? SYS_GPS : system;
00136     for (const auto& signal : ntrip_rtk_signals())
00137         {
00138             if (signal.system == lookup_system && !ntrip_rtk_is_entry_band(signal) &&
00139                 enabled.check_any_enabled(signal.flag))
00140                 {
00141                     return signal.band_slot;
00142                 }
00143         }
00144     // no enabled second band: the system's first listed second band from the
00145     // table, or the anticipated third slot for a constellation with no
00146     // second-band row yet (BeiDou B3I would live there)
00147     for (const auto& signal : ntrip_rtk_signals())
00148         {
00149             if (signal.system == lookup_system && !ntrip_rtk_is_entry_band(signal))
00150                 {
00151                     return signal.band_slot;
00152                 }
00153         }
00154     return 2;
```

```
00155 }
00156
00157
00158 //! Constellation label of an RTKLIB system bit, for messages
00159 inline const char* ntrip_rtk_system_label(int system)
00160 {
00161     switch (system)
00162         {
00163         case SYS_GPS:
00164             return "GPS";
00165         case SYS_GAL:
00166             return "Galileo";
00167         case SYS_BDS:
00168             return "BeiDou";
00169         case SYS_QZS:
00170             return "QZSS";
00171         default:
00172             return "unknown system";
00173         }
00174 }
00175
00176
00177 /*!
00178  * True when the enabled channel set follows the NTRIP RTK rules, all of
00179  * them derived from the table's columns: only signals the table lists, at
00180  * least one constellation entry band, at most one second band per system,
00181  * and every second band accompanied by its system's entry band.
00182  */
00183 inline bool ntrip_rtk_valid_channel_set(const Signal_Enabled_Flags& enabled)
00184 {
00185     if ((enabled.flags & ~ntrip_rtk_supported_signal_mask()) != 0)
00186         {
00187             return false;
00188         }
00189     bool any_first_band = false;
00190     for (const auto& signal : ntrip_rtk_signals())
00191         {
00192             if (!enabled.check_any_enabled(signal.flag))
00193                 {
00194                     continue;
00195                 }
00196             if (ntrip_rtk_is_entry_band(signal))
00197                 {
00198                     any_first_band = true;
00199                     continue;
00200                 }
00201             bool first_band_enabled = false;
00202             int enabled_second_bands = 0;
00203             for (const auto& sibling : ntrip_rtk_signals())
00204                 {
00205                     if (sibling.system != signal.system)
00206                         {
00207                             continue;
00208                         }
00209                     if (ntrip_rtk_is_entry_band(sibling))
00210                         {
00211                             first_band_enabled = first_band_enabled ||
00212                                                  enabled.check_any_enabled(sibling.flag);
00213                         }
00214                     else if (enabled.check_any_enabled(sibling.flag))
00215                         {
00216                             ++enabled_second_bands;
00217                         }
00218                 }
00219             if (!first_band_enabled || enabled_second_bands > 1)
00220                 {
00221                     return false;
00222                 }
00223         }
00224     return any_first_band;
00225 }
00226
00227
00228 //! Human-readable list of the supported per-system channel sets, derived
00229 //! from the table for the adapter's rejection message
00230 inline std::string ntrip_rtk_supported_sets_description()
00231 {
00232     std::string description;
00233     for (const auto& signal : ntrip_rtk_signals())
00234         {
00235             if (!ntrip_rtk_is_entry_band(signal))
00236                 {
00237                     continue;
00238                 }
00239             if (!description.empty())
00240                 {
00241                     description += "; ";
```

```
00242                 }
00243             description += ntrip_rtk_system_label(signal.system);
00244             description += ' ';
00245             description += signal.signal;
00246             std::string second_bands;
00247             for (const auto& sibling : ntrip_rtk_signals())
00248                 {
00249                     if (sibling.system != signal.system || ntrip_rtk_is_entry_band(sibling))
00250                         {
00251                             continue;
00252                         }
00253                     if (!second_bands.empty())
00254                         {
00255                             second_bands += " or ";
00256                         }
00257                     second_bands += sibling.signal;
00258                 }
00259             description += second_bands.empty() ? " alone" : (" alone or with one of " +
      second_bands);
00260         }
00261     return description;
00262 }
00263
00264
00265 /** \} */
00266 /** \} */
00267 #endif  // GNSS_SDR_NTRIP_RTK_SIGNALS_H
```

13.257 pvt_conf.h File Reference

Class that contains all the configuration parameters for the PVT block.

```
#include "secure_string.h"
#include <cstdint>
#include <map>
#include <string>
```

Classes

- class Pvt_Conf

13.257.1 Detailed Description

Class that contains all the configuration parameters for the PVT block.

Author

Carles Fernandez, 2018. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pvt_conf.h.

13.258 pvt_conf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pvt_conf.h
00003  * \brief Class that contains all the configuration parameters for the PVT block
00004  * \author Carles Fernandez, 2018. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_PVT_CONF_H
```

```
00018 #define GNSS_SDR_PVT_CONF_H
00019
00020 #include "secure_string.h"
00021 #include <cstdint>
00022 #include <map>
00023 #include <string>
00024
00025 /** \addtogroup PVT
00026  * \{ */
00027 /** \addtogroup PVT_libs
00028  * \{ */
00029
00030
00031 class Pvt_Conf
00032 {
00033 public:
00034     std::map<int, int> rtcm_msg_rate_ms;
00035
00036     std::string rinex_name = std::string("-");
00037     std::string dump_filename;
00038     std::string nmea_dump_filename;
00039     std::string nmea_dump_devname;
00040     std::string rtcm_dump_devname;
00041     std::string an_dump_devname;
00042     std::string output_path = std::string(".");
00043     std::string rinex_output_path = std::string(".");
00044     std::string gpx_output_path = std::string(".");
00045     std::string geojson_output_path = std::string(".");
00046     std::string nmea_output_file_path = std::string(".");
00047     std::string kml_output_path = std::string(".");
00048     std::string xml_output_path = std::string(".");
00049     std::string rtcm_output_file_path = std::string(".");
00050     std::string has_output_file_path = std::string(".");
00051     std::string udp_addresses;
00052     std::string udp_ports;
00053     std::string udp_eph_addresses;
00054     std::string log_source_timetag_file;
00055     std::string ntrip_caster_address;
00056     std::string ntrip_mountpoint;
00057     // Self-scrubbing: every copy of this struct wipes the caster credentials
00058     // on release, so new holders need no manual scrub site (the explicit
00059     // early scrubs via secure_clear_ntrip_credentials() remain worthwhile
00060     // for long-lived copies)
00061     Secure_String ntrip_username;
00062     Secure_String ntrip_password;
00063     Secure_String ntrip_password_env;
00064
00065     uint32_t signal_enabled_flags = 0;
00066     uint32_t observable_interval_ms = 20;
00067
00068     int32_t output_rate_ms = 0;
00069     int32_t display_rate_ms = 0;
00070     int32_t kml_rate_ms = 20;
00071     int32_t gpx_rate_ms = 20;
00072     int32_t geojson_rate_ms = 20;
00073     int32_t nmea_rate_ms = 20;
00074     int32_t rinex_version = 0;
00075     int32_t rinexobs_rate_ms = 0;
00076     int32_t an_rate_ms = 20;
00077     int32_t max_obs_block_rx_clock_offset_ms = 40;
00078     int32_t ref_gps_week = 0;  // reference GPS week to resolve the mod-1024 week rollover in
    post-processing (0: use system clock)
00079     int udp_eph_port = 0;
00080     int rtk_trace_level = 0;
00081     int32_t ntrip_timeout_ms = 10000;
00082     int32_t ntrip_reconnect_interval_ms = 10000;
00083     int32_t ntrip_version = 2;  // 2: prefer v2 with one compatible v1 retry; 1: force legacy v1
00084     int32_t ntrip_gga_period_ms = 10000;
00085
00086     uint16_t rtcm_tcp_port = 0;
00087     uint16_t rtcm_station_id = 0;
00088     uint16_t ntrip_port = 2101;
00089     uint16_t ntrip_station_id = 0;
00090
00091     double ntrip_max_correction_age_s = 5.0;
00092
00093     bool flag_nmea_tty_port = false;
00094     bool flag_rtcm_server = false;
00095     bool flag_rtcm_tty_port = false;
00096     bool output_enabled = true;
00097     bool rinex_output_enabled = true;
00098     bool gpx_output_enabled = true;
00099     bool geojson_output_enabled = true;
00100     bool nmea_output_file_enabled = true;
00101     bool an_output_enabled = false;
00102     bool kml_output_enabled = true;
00103     bool xml_output_enabled = true;
```

```
00104     bool rtcm_output_file_enabled = true;
00105     bool monitor_enabled = false;
00106     bool monitor_ephemeris_enabled = false;
00107     bool protobuf_enabled = true;
00108     bool enable_rx_clock_correction = true;
00109     bool show_local_time_zone = false;
00110     bool dump = false;
00111     bool dump_mat = true;
00112     bool log_source_timetag = false;
00113     bool ntrip_client_enabled = false;
00114     bool ntrip_fallback_to_single = true;
00115     bool ntrip_tls_enabled = false;
00116     bool ntrip_send_gga = true;  // VRS/nearest-station casters need the rover GGA to serve
      corrections
00117     bool use_e6_for_pvt = true;
00118     bool use_has_corrections = true;
00119     bool use_unhealthy_sats = false;
00120     bool glonass_strict_health = true;
00121     bool osnma_strict = false;
00122
00123     // PVT KF parameters
00124     bool enable_pvt_kf = false;
00125     double measures_ecef_pos_sd_m = 1.0;
00126     double measures_ecef_vel_sd_ms = 0.1;
00127     double system_ecef_pos_sd_m = 0.01;
00128     double system_ecef_vel_sd_ms = 0.001;
00129
00130     // Sensor Data parameters
00131     bool kf_use_imu_vel = false;
00132 };
00133
00134
00135 /** \} */
00136 /** \} */
00137 #endif  // GNSS_SDR_PVT_CONF_H
```

13.259 pvt_kf.h File Reference

Kalman Filter for Position and Velocity.

`#include <armadillo>`

Classes

- class Pvt_Kf

  Kalman Filter for Position and Velocity.

13.259.1 Detailed Description

Kalman Filter for Position and Velocity.

Author

Javier Arribas, 2023. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pvt_kf.h.

13.260 pvt_kf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pvt_kf.h
00003  * \brief Kalman Filter for Position and Velocity
00004  * \author Javier Arribas, 2023. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
```

```
00012  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_PVT_KF_H
00019 #define GNSS_SDR_PVT_KF_H
00020
00021 #include <armadillo>
00022
00023 /** \addtogroup PVT
00024  * \{ */
00025 /** \addtogroup PVT_libs
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief Kalman Filter for Position and Velocity
00031  *
00032  */
00033 class Pvt_Kf
00034 {
00035 public:
00036     Pvt_Kf() = default;
00037     virtual ~Pvt_Kf() = default;
00038     void init_Kf(const arma::vec& p,
00039         const arma::vec& v,
00040         double update_interval_s,
00041         double measures_ecef_pos_sd_m,
00042         double measures_ecef_vel_sd_ms,
00043         double system_ecef_pos_sd_m,
00044         double system_ecef_vel_sd_ms);
00045     bool is_initialized() const;
00046     void run_Kf(const arma::vec& p, const arma::vec& v);
00047     void get_pv_Kf(arma::vec& p, arma::vec& v) const;
00048     void reset_Kf();
00049
00050 private:
00051     // Kalman Filter class variables
00052     arma::mat d_F;
00053     arma::mat d_H;
00054     arma::mat d_R;
00055     arma::mat d_Q;
00056     arma::mat d_P_old_old;
00057     arma::mat d_P_new_old;
00058     arma::mat d_P_new_new;
00059     arma::vec d_x_old_old;
00060     arma::vec d_x_new_old;
00061     arma::vec d_x_new_new;
00062     bool d_initialized{false};
00063 };
00064
00065
00066 /** \} */
00067 /** \} */
00068 #endif  // GNSS_SDR_Pvt_Kf_H
```

13.261 pvt_solution.h File Reference

Interface of a base class for a PVT solution.

```
#include <boost/date_time/posix_time/posix_time.hpp>
#include <array>
#include <cstdint>
```

Classes

- class Pvt_Solution

  Base class for a PVT solution.

13.261.1 Detailed Description

Interface of a base class for a PVT solution.

Author

Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pvt_solution.h.

13.262 pvt_solution.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pvt_solution.h
00003  * \brief Interface of a base class for a PVT solution
00004  * \author Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_PVT_SOLUTION_H
00019 #define GNSS_SDR_PVT_SOLUTION_H
00020
00021 #include <boost/date_time/posix_time/posix_time.hpp>
00022 #include <array>
00023 #include <cstdint>
00024
00025 /** \addtogroup PVT
00026  * \{ */
00027 /** \addtogroup PVT_libs
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief Base class for a PVT solution
00033  *
00034  */
00035 class Pvt_Solution
00036 {
00037 public:
00038     Pvt_Solution() = default;
00039     virtual ~Pvt_Solution() = default;
00040
00041     virtual double get_hdop() const = 0;
00042     virtual double get_vdop() const = 0;
00043     virtual double get_pdop() const = 0;
00044     virtual double get_gdop() const = 0;
00045
00046     std::array<double, 3> get_rx_pos() const;
00047     std::array<double, 3> get_rx_vel() const;
00048     boost::posix_time::ptime get_position_UTC_time() const;
00049     double get_latitude() const;             //!< Get RX position Latitude WGS84 [deg]
00050     double get_longitude() const;            //!< Get RX position Longitude WGS84 [deg]
00051     double get_height() const;               //!< Get RX position height WGS84 [m]
00052     double get_time_offset_s() const;        //!< Get RX time offset [s]
00053     double get_clock_drift_ppm() const;      //!< Get the Rx clock drift [ppm]
00054     double get_speed_over_ground() const;    //!< Get RX speed over ground [m/s]
00055     double get_course_over_ground() const;   //!< Get RX course over ground [deg]
00056     int get_num_valid_observations() const;  //!< Get the number of valid pseudorange observations
      (valid satellites)
00057     int32_t get_ref_gps_week() const;        //!< Get the reference GPS week used to resolve the
      mod-1024 week rollover (0 if unset)
00058     bool is_valid_position() const;
00059
00060     void set_rx_pos(const std::array<double, 3> &pos);  //!< Set position: X, Y, Z in Cartesian ECEF
      coordinates [m]
00061     void set_rx_vel(const std::array<double, 3> &vel);  //!< Set velocity: East [m/s], North [m/s], Up
      [m/s]
00062     void set_position_UTC_time(const boost::posix_time::ptime &pt);
00063     void set_time_offset_s(double offset);             //!< Set RX time offset [s]
00064     void set_clock_drift_ppm(double clock_drift_ppm);  //!< Set the Rx clock drift [ppm]
00065     void set_speed_over_ground(double speed_m_s);      //!< Set RX speed over ground [m/s]
00066     void set_course_over_ground(double cog_deg);       //!< Set RX course over ground [deg]
00067     void set_valid_position(bool is_valid);
```

```
00068     void set_num_valid_observations(int num);  //!< Set the number of valid pseudorange observations
      (valid satellites)
00069     void set_ref_gps_week(int32_t ref_week);   //!< Set the reference GPS week (e.g., derived from the
      date of signal capture) used to resolve the mod-1024 week rollover in post-processing mode
00070
00071 private:
00072     /*
00073      * Conversion of Cartesian coordinates (X,Y,Z) to geographical
00074      * coordinates (d_latitude_d, d_longitude_d, d_height_m) on a selected reference ellipsoid.
00075      *
00076      * \param[in] X [m] Cartesian coordinate
00077      * \param[in] Y [m] Cartesian coordinate
00078      * \param[in] Z [m] Cartesian coordinate
00079      * \param[in] elipsoid_selection. Choices of Reference Ellipsoid for Geographical Coordinates:
00080      * 0 - International Ellipsoid 1924.
00081      * 1 - International Ellipsoid 1967.
00082      * 2 - World Geodetic System 1972.
00083      * 3 - Geodetic Reference System 1980.
00084      * 4 - World Geodetic System 1984.
00085      *
00086      */
00087     int cart2geo(double X, double Y, double Z, int elipsoid_selection);
00088
00089     std::array<double, 3> d_rx_pos{};
00090     std::array<double, 3> d_rx_vel{};
00091     boost::posix_time::ptime d_position_UTC_time;
00092
00093     double d_latitude_d{0.0};             // RX position Latitude WGS84 [deg]
00094     double d_longitude_d{0.0};            // RX position Longitude WGS84 [deg]
00095     double d_height_m{0.0};               // RX position height WGS84 [m]
00096     double d_rx_dt_s{0.0};                // RX time offset [s]
00097     double d_rx_clock_drift_ppm{0.0};     // RX clock drift [ppm]
00098     double d_speed_over_ground_m_s{0.0};  // RX speed over ground [m/s]
00099     double d_course_over_ground_d{0.0};   // RX course over ground [deg]
00100
00101     int d_valid_observations{0};  // Number of valid observations in this epoch
00102
00103     int32_t d_ref_gps_week{0};  // Reference GPS week to correct the mod-1024 week rollover in
      post-processing mode (0: use system clock)
00104
00105     bool d_valid_position{false};
00106 };
00107
00108
00109 /** \} */
00110 /** \} */
00111 #endif  // GNSS_SDR_PVT_SOLUTION_H
```

13.263 rinex_printer.h File Reference

Interface of a RINEX 2.11 / 3.02 / 4.02 printer See https://files.igs.org/pub/data/format/↩rinex_4.02.pdf.

```
#include "signal_enabled_flags.h"
#include <boost/date_time/posix_time/posix_time.hpp>
#include <cstdint>
#include <cstdlib>
#include <fstream>
#include <map>
#include <string>
#include <vector>
```

Classes

- class Rinex_Printer

  Class that handles the generation of Receiver INdependent EXchange format (RINEX) files.

13.263.1 Detailed Description

Interface of a RINEX 2.11 / 3.02 / 4.02 printer See https://files.igs.org/pub/data/format/↩rinex_4.02.pdf.

Receiver Independent EXchange Format (RINEX): The first proposal for the Receiver Independent Exchange Format RINEX was developed by the Astronomical Institute of the University of Berne for the easy exchange of the

GPS data to be collected during the large European GPS campaign EUREF 89, which involved more than 60 GPS receivers of 4 different manufacturers. The governing aspect during the development was the fact that most geodetic processing software for GPS data use a well-defined set of observables: 1) The carrier-phase measurement at one or both carriers (actually being a measurement on the beat frequency between the received carrier of the satellite signal and a receiver-generated reference frequency). 2) The pseudorange (code) measurement , equivalent to the difference of the time of reception (expressed in the time frame of the receiver) and the time of transmission (expressed in the time frame of the satellite) of a distinct satellite signal. 3) The observation time being the reading of the receiver clock at the instant of validity of the carrier-phase and/or the code measurements. Note: A collection of the formats currently used by the IGS can be found here: https://igs.org/formats-and-standards/

Author

Carles Fernandez Prades, 2011-2026. cfernandez(at)cttc.es

Mathieu Favreau, 2025-2026. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file rinex_printer.h.

13.264 rinex_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rinex_printer.h
00003  * \brief Interface of a RINEX 2.11 / 3.02 / 4.02 printer
00004  * See https://files.igs.org/pub/data/format/rinex_4.02.pdf
00005  *
00006  * Receiver Independent EXchange Format (RINEX):
00007  * The first proposal for the Receiver Independent Exchange Format RINEX
00008  * was developed by the Astronomical Institute of the University of Berne
00009  * for the easy exchange of the GPS data to be collected during the large
00010  * European GPS campaign EUREF 89, which involved more than 60 GPS receivers
00011  * of 4 different manufacturers.
00012  * The governing aspect during the development was the fact that most geodetic
00013  * processing software for GPS data use a well-defined set of observables:
00014  * 1) The carrier-phase measurement at one or both carriers (actually being a
00015  * measurement on the beat frequency between the received carrier of the
00016  * satellite signal and a receiver-generated reference frequency).
00017  * 2) The pseudorange (code) measurement , equivalent to the difference
00018  * of the time of reception (expressed in the time frame of the receiver)
00019  * and the time of transmission (expressed in the time frame of the satellite)
00020  * of a distinct satellite signal.
00021  * 3) The observation time being the reading of the receiver clock at the
00022  * instant of validity of the carrier-phase and/or the code measurements.
00023  * Note: A collection of the formats currently used by the IGS can be found
00024  * here: https://igs.org/formats-and-standards/
00025  * \author Carles Fernandez Prades, 2011-2026. cfernandez(at)cttc.es
00026  * \author Mathieu Favreau, 2025-2026. favreau.mathieu(at)hotmail.com
00027  *
00028  * -----------------------------------------------------------------------------
00029  *
00030  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00031  * This file is part of GNSS-SDR.
00032  *
00033  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00034  * SPDX-License-Identifier: GPL-3.0-or-later
00035  *
00036  * -----------------------------------------------------------------------------
00037  */
00038
00039 #ifndef GNSS_SDR_RINEX_PRINTER_H
00040 #define GNSS_SDR_RINEX_PRINTER_H
00041
00042 #include "signal_enabled_flags.h"
00043 #include <boost/date_time/posix_time/posix_time.hpp>
00044 #include <cstdint>  // for int32_t
00045 #include <cstdlib>  // for strtol, strtod
00046 #include <fstream>  // for fstream
00047 #include <map>      // for map
00048 #include <string>   // for string
00049 #include <vector>   // for vector
00050
00051
00052 /** \addtogroup PVT
00053  * \{ */
00054 /** \addtogroup PVT_libs
00055  * \{ */
```

```
00056
00057
00058 struct Bds3_B1c_PageData;
00059 class Beidou_Cnav1_Ephemeris;
00060 class Beidou_Dnav_Ephemeris;
00061 class Beidou_Dnav_Iono;
00062 class Beidou_Dnav_Utc_Model;
00063 class Galileo_Ephemeris;
00064 class Galileo_Iono;
00065 class Galileo_Utc_Model;
00066 class Glonass_Gnav_Almanac;
00067 class Glonass_Gnav_Ephemeris;
00068 class Glonass_Gnav_Utc_Model;
00069 class Gnss_Synchro;
00070 class Gps_CNAV_Ephemeris;
00071 class Gps_CNAV_Iono;
00072 class Gps_CNAV_Utc_Model;
00073 class Gps_Ephemeris;
00074 class Gps_Iono;
00075 class Gps_Navigation_Message;
00076 class Gps_Utc_Model;
00077 class Rtklib_Solver;
00078
00079
00080 /*!
00081  * \brief Class that handles the generation of Receiver
00082  * INdependent EXchange format (RINEX) files
00083  */
00084 class Rinex_Printer
00085 {
00086 public:
00087     /*!
00088      * \brief Constructor. Creates GNSS Navigation and Observables RINEX files.
00089      */
00090     explicit Rinex_Printer(uint32_t signal_enabled_flags,
00091         int version = 3,
00092         const std::string& base_path = ".",
00093         const std::string& base_name = "-",
00094         int32_t ref_gps_week = 0);
00095
00096     /*!
00097      * \brief Destructor. Removes created files if empty.
00098      */
00099     ~Rinex_Printer() noexcept;
00100
00101     /*!
00102      * \brief Print RINEX annotation. If it is the first annotation, it also
00103      * prints the RINEX headers for navigation and observation files. If it is
00104      * not the first annotation, it only annotates the observation, and updates
00105      * the navigation header if UTC data was not available when writing it for
00106      * the first time.
00107      *
00108      */
00109     void print_rinex_annotation(const Rtklib_Solver* pvt_solver,
00110         const std::map<int, Gnss_Synchro>& gnss_observables_map,
00111         double rx_time,
00112         bool flag_write_RINEX_obs_output);
00113
00114     /*!
00115      * \brief Print RINEX annotation for GPS NAV message
00116      */
00117     void log_rinex_nav_gps_nav(const std::map<int32_t, Gps_Ephemeris>& new_eph);
00118
00119     /*!
00120      * \brief Print RINEX 4 annotation for GPS/QZSS CNAV message
00121      */
00122     void log_rinex_nav_gps_cnav(const std::map<int32_t, Gps_CNAV_Ephemeris>& new_cnav_eph);
00123
00124     /*!
00125      * \brief Print RINEX annotation for Galileo NAV message
00126      */
00127     void log_rinex_nav_gal_nav(const std::map<int32_t, Galileo_Ephemeris>& new_gal_eph);
00128
00129     /*!
00130      * \brief Print RINEX annotation for Glonass GNAV message
00131      */
00132     void log_rinex_nav_glo_gnav(const std::map<int32_t, Glonass_Gnav_Ephemeris>& new_glo_eph);
00133
00134     /*!
00135      * \brief Print RINEX annotation for BeiDou DNAV message
00136      */
00137     void log_rinex_nav_bds_dnav(const std::map<int32_t, Beidou_Dnav_Ephemeris>& new_bds_eph);
00138
00139     /*!
00140      * \brief Print RINEX navigation records for BeiDou B-CNAV1 ephemerides.
00141      *
00142      * RINEX 4 writes proper CNV1 records (Table A24), taking SISAI/SISMAI/
```

```
00143      * integrity and t_op from the last decoded SF3 page when it carries them.
00144      * RINEX 3 has no CNAV1 representation, so records are written as D1-style
00145      * stand-ins: sqrt(A0) in the sqrtA field with the Adot term dropped,
00146      * IODE/IODC in AODE/AODC, and TGD_B1Cp/TGD_B2ap mapped into TGD1/TGD2.
00147      */
00148     void log_rinex_nav_bds_cnav1(const std::map<int32_t, Beidou_Cnav1_Ephemeris>& new_bds_eph,
00149         const std::map<int32_t, Bds3_B1c_PageData>& bds_page_data);
00150
00151     /*!
00152      * \brief Print RINEX navigation records for BeiDou B-CNAV2 ephemerides.
00153      *
00154      * RINEX 4 writes CNV2 records (Table A25) with ISC_B2ad in the second field of orbit line 7.
00155      * RINEX 3 has no CNV2 representation; records are omitted rather than
00156      * written as CNV1 or D1 stand-ins.
00157      */
00158     void log_rinex_nav_bds_cnav2(const std::map<int32_t, Beidou_Cnav1_Ephemeris>& new_bds_eph);
00159
00160     /*!
00161      * \brief Returns true is the RINEX file headers are already written
00162      */
00163     inline bool is_rinex_header_written() const
00164     {
00165         return d_rinex_header_written;
00166     }
00167
00168     /*!
00169      * \brief Returns name of RINEX navigation file(s)
00170      */
00171     inline std::vector<std::string> get_navfilename() const
00172     {
00173         return output_navfilename;
00174     }
00175
00176     /*!
00177      * \brief Returns name of RINEX observation file
00178      */
00179     inline std::string get_obsfilename() const
00180     {
00181         return obsfilename;
00182     }
00183
00184
00185 private:
00186     // Not the best, but reorder params to select the correct constructor
00187     explicit Rinex_Printer(uint32_t signal_enabled_flags,
00188         const std::string& base_name,
00189         const std::string& base_rinex_path,
00190         int version,
00191         int32_t ref_gps_week);
00192
00193     /*
00194      * Generates the GPS Observation data header
00195      */
00196     void rinex_obs_header(std::fstream& out,
00197         const std::string& time_constellation,
00198         const boost::posix_time::ptime& system_time,
00199         double seconds);
00200
00201     /*
00202      * Generates the Navigation Data header
00203      */
00204     void rinex_nav_header(std::fstream& out,
00205         const std::vector<std::string>& iono_lines,
00206         const std::vector<std::string>& time_corr_lines,
00207         const std::string& leap_second_line) const;
00208
00209     /*
00210      * Logs RINEX 4 auxiliary data records for the enabled constellations as
00211      * soon as they become available, and updates the LEAP SECONDS header line
00212      * of the observation and navigation files
00213      */
00214     void log_rinex_nav_v4_aux_records(const Rtklib_Solver* pvt_solver,
00215         double rx_time,
00216         const std::string& system_time_str);
00217
00218     /*
00219      * Computes the BDS Time and returns a boost::posix_time::ptime object
00220      *  \details Function used to convert the observation time into BDT time which is used
00221      *  as the default time for RINEX files
00222      *  \param eph BeiDou DNAV Ephemeris object
00223      *  \param obs_time Observation time in BDT seconds of week
00224      */
00225     boost::posix_time::ptime compute_BDS_time(const Beidou_Dnav_Ephemeris& eph, double obs_time)
    const;
00226     boost::posix_time::ptime compute_BDS_time(const Beidou_Cnav1_Ephemeris& eph, double obs_time)
    const;
00227
```

```
00228     /*
00229      * Computes the UTC time and returns a boost::posix_time::ptime object
00230      */
00231     boost::posix_time::ptime compute_UTC_time(const Gps_Navigation_Message& nav_msg) const;
00232
00233     /*
00234      * Computes the GPS time and returns a boost::posix_time::ptime object
00235      */
00236     boost::posix_time::ptime compute_GPS_time(const Gps_Ephemeris& eph, double obs_time) const;
00237
00238     /*
00239      * Computes the GPS time and returns a boost::posix_time::ptime object
00240      */
00241     boost::posix_time::ptime compute_GPS_time(const Gps_CNAV_Ephemeris& eph, double obs_time) const;
00242
00243     /*
00244      * Computes the Galileo time and returns a boost::posix_time::ptime object
00245      */
00246     boost::posix_time::ptime compute_Galileo_time(const Galileo_Ephemeris& eph, double obs_time)
      const;
00247
00248     /*
00249      * Computes the UTC Time and returns a boost::posix_time::ptime object
00250      *  \details Function used as a method to convert the observation time into UTC time which is used
00251      *  as the default time for RINEX files
00252      *  \param eph GLONASS GNAV Ephemeris object
00253      *  \param obs_time Observation time in GPS seconds of week
00254      */
00255     boost::posix_time::ptime compute_UTC_time(const Glonass_Gnav_Ephemeris& eph, double obs_time)
      const;
00256
00257     /*
00258      * Represents GPS time in the date time format. Leap years are considered, but leap seconds are
      not.
00259      */
00260     void to_date_time(int gps_week,
00261         int gps_tow,
00262         int& year,
00263         int& month,
00264         int& day,
00265         int& hour,
00266         int& minute,
00267         int& second) const;
00268
00269     void update_obs_header(std::fstream& out, const std::string& leap_second_line) const;
00270
00271     const std::map<std::string, std::string> observationType;  // PSEUDORANGE, CARRIER_PHASE, DOPPLER,
      SIGNAL_STRENGTH
00272     const std::map<std::string, std::string> observationCode;  // GNSS observation descriptors
00273
00274     const Signal_Enabled_Flags d_flags;
00275
00276     const int d_version;                // RINEX version (2 for 2.11, 3 for 3.02, 4 for 4.02)
00277     const std::string d_stringVersion;  // RINEX version (2.11, 3.02 or 4.02)
00278
00279     std::string d_last_gps_lnav_iono_signature;
00280     std::string d_last_gps_lnav_sto_signature;
00281     std::string d_last_gps_cnav_eop_signature;
00282     std::string d_last_gps_cnav_iono_signature;
00283     std::string d_last_gps_cnav_sto_signature;
00284     std::string d_last_qzss_lnav_iono_signature;
00285     std::string d_last_qzss_lnav_sto_signature;
00286     std::string d_last_qzss_cnav_eop_signature;
00287     std::string d_last_qzss_cnav_iono_signature;
00288     std::string d_last_qzss_cnav_sto_signature;
00289     std::string d_last_galileo_iono_signature;
00290     std::string d_last_galileo_sto_signature;
00291     std::string d_last_glonass_sto_signature;
00292     std::string d_last_beidou_iono_signature;
00293     std::string d_last_beidou_sto_signature;
00294     std::string d_last_bds_cnav1_iono_signature;
00295     std::string d_last_bds_cnav1_sto_signature;
00296     std::string d_last_leap_second_line;
00297     const int32_t d_ref_gps_week;
00298
00299     const std::string navfilename;                // Name of RINEX navigation file
00300     const std::string obsfilename;                // Name of RINEX observation file
00301     const std::string navGlofilename;             // Name of RINEX navigation file for Glonass
00302     std::vector<std::string> output_navfilename;  // Name of output RINEX navigation file(s)
00303
00304     std::fstream obsFile;     // Output file stream for RINEX observation file
00305     std::fstream navFile;     // Output file stream for RINEX navigation data file
00306     std::fstream navGloFile;  // Output file stream for RINEX GLONASS navigation data file
00307
00308     bool d_rinex_header_updated;
00309     bool d_rinex_header_gps_updated;
00310     bool d_rinex_header_galileo_updated;
```

```
00311     bool d_rinex_header_glonass_updated;
00312     bool d_rinex_header_beidou_iono_updated;
00313     bool d_rinex_header_beidou_time_updated;
00314     bool d_rinex_header_written;
00315 };
00316
00317
00318 /** \} */
00319 /** \} */
00320 #endif  // GNSS_SDR_RINEX_PRINTER_H
```

13.265 rtcm.h File Reference

Interface for the RTCM 3.2 Standard.

```
#include "concurrent_queue.h"
#include "galileo_ephemeris.h"
#include "galileo_has_data.h"
#include "glonass_gnav_ephemeris.h"
#include "glonass_gnav_utc_model.h"
#include "gnss_synchro.h"
#include "gps_cnav_ephemeris.h"
#include "gps_ephemeris.h"
#include <boost/asio.hpp>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <algorithm>
#include <array>
#include <bitset>
#include <cstddef>
#include <cstdint>
#include <deque>
#include <iomanip>
#include <list>
#include <map>
#include <memory>
#include <set>
#include <sstream>
#include <string>
#include <thread>
#include <utility>
#include <vector>
#include <absl/log/log.h>
```

Classes

- class Rtcm

  This class implements the generation and reading of some Message Types defined in the RTCM 3.2 Standard, plus some utilities to handle messages.

Typedefs

- using b_io_context = boost::asio::io_service

13.265.1 Detailed Description

Interface for the RTCM 3.2 Standard.

Author

Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file rtcm.h.

13.266 rtcm.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtcm.h
00003  * \brief  Interface for the RTCM 3.2 Standard
00004  * \author Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_RTCM_H
00019 #define GNSS_SDR_RTCM_H
00020
00021
00022 #include "concurrent_queue.h"
00023 #include "galileo_ephemeris.h"
00024 #include "galileo_has_data.h"
00025 #include "glonass_gnav_ephemeris.h"
00026 #include "glonass_gnav_utc_model.h"
00027 #include "gnss_synchro.h"
00028 #include "gps_cnav_ephemeris.h"
00029 #include "gps_ephemeris.h"
00030 #include <boost/asio.hpp>
00031 #include <boost/date_time/posix_time/posix_time.hpp>
00032 #include <algorithm>  // for std::max, std::min, std::copy_n
00033 #include <array>
00034 #include <bitset>
00035 #include <cstddef>  // for size_t
00036 #include <cstdint>
00037 #include <deque>
00038 #include <iomanip>  // for std::setw
00039 #include <list>
00040 #include <map>
00041 #include <memory>
00042 #include <set>
00043 #include <sstream>  // for std::stringstream
00044 #include <string>
00045 #include <thread>
00046 #include <utility>
00047 #include <vector>
00048
00049 #if USE_GLOG_AND_GFLAGS
00050 #include <glog/logging.h>
00051 #else
00052 #include <absl/log/log.h>
00053 #endif
00054
00055 /** \addtogroup PVT
00056  * \{ */
00057 /** \addtogroup PVT_libs
00058  * \{ */
00059
00060
00061 #if USE_BOOST_ASIO_IO_CONTEXT
00062 using b_io_context = boost::asio::io_context;
00063 #else
00064 using b_io_context = boost::asio::io_service;
00065 #endif
00066
00067
00068 /*!
00069  * \brief This class implements the generation and reading of some Message Types
00070  * defined in the RTCM 3.2 Standard, plus some utilities to handle messages.
00071  *
00072  * Generation of the following Message Types:
```

```
00073  *   1001, 1002, 1003, 1004, 1005, 1006, 1008, 1019, 1020, 1029, 1045,
00074  *   1057, 1058, 1059, 1060
00075  *
00076  * Decoding of the following Message Types:
00077  *   1019, 1045
00078  *
00079  * Generation of the following Multiple Signal Messages:
00080  *   MSM1 (message types 1071, 1091)
00081  *   MSM2 (message types 1072, 1092)
00082  *   MSM3 (message types 1073, 1093)
00083  *   MSM4 (message types 1074, 1094)
00084  *   MSM5 (message types 1075, 1095)
00085  *   MSM6 (message types 1076, 1096)
00086  *   MSM7 (message types 1077, 1097)
00087  *
00088  * RTCM 3 message format (size in bits):
00089  *   +----------+--------+-----------+--------------------+----------+
00090  *   | preamble | 000000 |  length   |    data message    |  parity  |
00091  *   +----------+--------+-----------+--------------------+----------+
00092  *   |<-- 8 --->|<- 6 -->|<-- 10 --->|<--- length x 8 --->|<-- 24 -->|
00093  *   +----------+--------+-----------+--------------------+----------+
00094  *
00095  *
00096  *   (C) Carles Fernandez-Prades, 2015. cfernandez(at)cttc.es
00097  */
00098 class Rtcm
00099 {
00100 public:
00101     explicit Rtcm(uint16_t port = 2101);  //!< Default constructor that sets TCP port of the RTCM
      message server and RTCM Station ID. 2101 is the standard RTCM port according to the Internet Assigned
      Numbers Authority (IANA). See
      https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml
00102     ~Rtcm() noexcept;
00103
00104     /*!
00105      * \brief Prints message type 1001 (L1-Only GPS RTK Observables)
00106      */
00107     std::string print_MT1001(const Gps_Ephemeris& gps_eph, double obs_time, const std::map<int32_t,
      Gnss_Synchro>& observables, uint16_t station_id);
00108
00109     /*!
00110      * \brief Prints message type 1002 (Extended L1-Only GPS RTK Observables)
00111      */
00112     std::string print_MT1002(const Gps_Ephemeris& gps_eph, double obs_time, const std::map<int32_t,
      Gnss_Synchro>& observables, uint16_t station_id);
00113
00114     /*!
00115      * \brief Prints message type 1003 (L1 & L2 GPS RTK Observables)
00116      */
00117     std::string print_MT1003(const Gps_Ephemeris& ephL1, const Gps_CNAV_Ephemeris& ephL2, double
      obs_time, const std::map<int32_t, Gnss_Synchro>& observables, uint16_t station_id);
00118
00119     /*!
00120      * \brief Prints message type 1004 (Extended L1 & L2 GPS RTK Observables)
00121      */
00122     std::string print_MT1004(const Gps_Ephemeris& ephL1, const Gps_CNAV_Ephemeris& ephL2, double
      obs_time, const std::map<int32_t, Gnss_Synchro>& observables, uint16_t station_id);
00123
00124     /*!
00125      * \brief Prints message type 1005 (Stationary Antenna Reference Point)
00126      */
00127     std::string print_MT1005(uint32_t ref_id, double ecef_x, double ecef_y, double ecef_z, bool gps,
      bool glonass, bool galileo, bool non_physical, bool single_oscillator, uint32_t
      quarter_cycle_indicator);
00128
00129     /*!
00130      * \brief Verifies and reads messages of type 1005 (Stationary Antenna Reference Point). Returns 1
      if anything goes wrong, 0 otherwise.
00131      */
00132     int32_t read_MT1005(const std::string& message, uint32_t& ref_id, double& ecef_x, double& ecef_y,
      double& ecef_z, bool& gps, bool& glonass, bool& galileo);
00133
00134     /*!
00135      * \brief Prints message type 1006 (Stationary Antenna Reference Point, with Height Information)
00136      */
00137     std::string print_MT1006(uint32_t ref_id, double ecef_x, double ecef_y, double ecef_z, bool gps,
      bool glonass, bool galileo, bool non_physical, bool single_oscillator, uint32_t
      quarter_cycle_indicator, double height);
00138
00139     std::string print_MT1005_test();  //!< For testing purposes
00140
00141     /*!
00142      * \brief Prints message type 1008 (Antenna Descriptor & Serial Number)
00143      */
00144     std::string print_MT1008(uint32_t ref_id, const std::string& antenna_descriptor, uint32_t
      antenna_setup_id, const std::string& antenna_serial_number);
00145
```

```
00146     /*!
00147      * \brief Prints L1-Only GLONASS RTK Observables
00148      * \details This GLONASS message type is not generally used or supported; type 1012 is to be
      preferred.
00149      * \note Code added as part of GSoC 2017 program
00150      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00151      * \param obs_time Time of observation at the moment of printing
00152      * \param observables Set of observables as defined by the platform
00153      * \return string with message contents
00154      */
00155     std::string print_MT1009(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, double obs_time, const
      std::map<int32_t, Gnss_Synchro>& observables, uint16_t station_id);
00156
00157     /*!
00158      * \brief Prints Extended L1-Only GLONASS RTK Observables
00159      * \details This GLONASS message type is used when only L1 data is present and bandwidth is very
      tight, often 1012 is used in such cases.
00160      * \note Code added as part of GSoC 2017 program
00161      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00162      * \param obs_time Time of observation at the moment of printing
00163      * \param observables Set of observables as defined by the platform
00164      * \return string with message contents
00165      */
00166     std::string print_MT1010(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, double obs_time, const
      std::map<int32_t, Gnss_Synchro>& observables, uint16_t station_id);
00167
00168     /*!
00169      * \brief Prints L1&L2 GLONASS RTK Observables
00170      * \details This GLONASS message type is not generally used or supported; type 1012 is to be
      preferred
00171      * \note Code added as part of GSoC 2017 program
00172      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00173      * \param obs_time Time of observation at the moment of printing
00174      * \param observables Set of observables as defined by the platform
00175      * \return string with message contents
00176      */
00177     std::string print_MT1011(const Glonass_Gnav_Ephemeris& glonass_gnav_ephL1, const
      Glonass_Gnav_Ephemeris& glonass_gnav_ephL2, double obs_time, const std::map<int32_t, Gnss_Synchro>&
      observables, uint16_t station_id);
00178
00179     /*!
00180      * \brief Prints Extended L1&L2 GLONASS RTK Observables
00181      * \details This GLONASS message type is the most common observational message type, with
      L1/L2/SNR content.  This is one of the most common messages found.
00182      * \note Code added as part of GSoC 2017 program
00183      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00184      * \param obs_time Time of observation at the moment of printing
00185      * \param observables Set of observables as defined by the platform
00186      * \return string with message contents
00187      */
00188     std::string print_MT1012(const Glonass_Gnav_Ephemeris& glonass_gnav_ephL1, const
      Glonass_Gnav_Ephemeris& glonass_gnav_ephL2, double obs_time, const std::map<int32_t, Gnss_Synchro>&
      observables, uint16_t station_id);
00189
00190     /*!
00191      * \brief Prints message type 1019 (GPS Ephemeris), should be broadcast in the event that
00192      * the IODC does not match the IODE, and every 2 minutes.
00193      */
00194     std::string print_MT1019(const Gps_Ephemeris& gps_eph);
00195
00196     /*!
00197      * \brief Verifies and reads messages of type 1019 (GPS Ephemeris). Returns 1 if anything goes
      wrong, 0 otherwise.
00198      */
00199     int32_t read_MT1019(const std::string& message, Gps_Ephemeris& gps_eph) const;
00200
00201     /*!
00202      * \brief Prints message type 1020 (GLONASS Ephemeris).
00203      * \note Code added as part of GSoC 2017 program
00204      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00205      * \param glonass_gnav_utc_model GLONASS GNAV Clock Information
00206      * \return Returns message type as a string type
00207      */
00208     std::string print_MT1020(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, const
      Glonass_Gnav_Utc_Model& glonass_gnav_utc_model);
00209
00210     /*!
00211      * \brief Verifies and reads messages of type 1020 (GLONASS Ephemeris).
00212      * \note Code added as part of GSoC 2017 program
00213      * \param message Message to read as a string type
00214      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00215      * \param glonass_gnav_utc_model GLONASS GNAV Clock Information
00216      * \return Returns 1 if anything goes wrong, 0 otherwise.
00217      */
00218     int32_t read_MT1020(const std::string& message, Glonass_Gnav_Ephemeris& glonass_gnav_eph,
      Glonass_Gnav_Utc_Model& glonass_gnav_utc_model) const;
00219
```

```
00220     /*!
00221      * \brief Prints message type 1029 (Unicode Text String)
00222      */
00223     std::string print_MT1029(uint32_t ref_id, const Gps_Ephemeris& gps_eph, double obs_time, const
      std::string& message);
00224
00225     /*!
00226      * \brief Prints message type 1045 (Galileo Ephemeris), should be broadcast every 2 minutes
00227      */
00228     std::string print_MT1045(const Galileo_Ephemeris& gal_eph);
00229
00230     /*!
00231      * \brief Verifies and reads messages of type 1045 (Galileo Ephemeris). Returns 1 if anything goes
      wrong, 0 otherwise.
00232      */
00233     int32_t read_MT1045(const std::string& message, Galileo_Ephemeris& gal_eph) const;
00234
00235     /*!
00236      * \brief Prints messages of type MSM1 (Compact GNSS observables)
00237      */
00238     std::string print_MSM_1(const Gps_Ephemeris& gps_eph,
00239         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00240         const Galileo_Ephemeris& gal_eph,
00241         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00242         double obs_time,
00243         const std::map<int32_t, Gnss_Synchro>& observables,
00244         uint32_t ref_id,
00245         uint32_t clock_steering_indicator,
00246         uint32_t external_clock_indicator,
00247         int32_t smooth_int,
00248         bool divergence_free,
00249         bool more_messages);
00250
00251     /*!
00252      * \brief Prints messages of type MSM2 (Compact GNSS phaseranges)
00253      */
00254     std::string print_MSM_2(const Gps_Ephemeris& gps_eph,
00255         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00256         const Galileo_Ephemeris& gal_eph,
00257         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00258         double obs_time,
00259         const std::map<int32_t, Gnss_Synchro>& observables,
00260         uint32_t ref_id,
00261         uint32_t clock_steering_indicator,
00262         uint32_t external_clock_indicator,
00263         int32_t smooth_int,
00264         bool divergence_free,
00265         bool more_messages);
00266
00267     /*!
00268      * \brief Prints messages of type MSM3 (Compact GNSS pseudoranges and phaseranges)
00269      */
00270     std::string print_MSM_3(const Gps_Ephemeris& gps_eph,
00271         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00272         const Galileo_Ephemeris& gal_eph,
00273         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00274         double obs_time,
00275         const std::map<int32_t, Gnss_Synchro>& observables,
00276         uint32_t ref_id,
00277         uint32_t clock_steering_indicator,
00278         uint32_t external_clock_indicator,
00279         int32_t smooth_int,
00280         bool divergence_free,
00281         bool more_messages);
00282
00283     /*!
00284      * \brief Prints messages of type MSM4 (Full GNSS pseudoranges and phaseranges plus CNR)
00285      */
00286     std::string print_MSM_4(const Gps_Ephemeris& gps_eph,
00287         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00288         const Galileo_Ephemeris& gal_eph,
00289         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00290         double obs_time,
00291         const std::map<int32_t, Gnss_Synchro>& observables,
00292         uint32_t ref_id,
00293         uint32_t clock_steering_indicator,
00294         uint32_t external_clock_indicator,
00295         int32_t smooth_int,
00296         bool divergence_free,
00297         bool more_messages);
00298
00299     /*!
00300      * \brief Prints messages of type MSM5 (Full GNSS pseudoranges, phaseranges, phaserange rate and
      CNR)
00301      */
00302     std::string print_MSM_5(const Gps_Ephemeris& gps_eph,
00303         const Gps_CNAV_Ephemeris& gps_cnav_eph,
```

```
00304         const Galileo_Ephemeris& gal_eph,
00305         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00306         double obs_time,
00307         const std::map<int32_t, Gnss_Synchro>& observables,
00308         uint32_t ref_id,
00309         uint32_t clock_steering_indicator,
00310         uint32_t external_clock_indicator,
00311         int32_t smooth_int,
00312         bool divergence_free,
00313         bool more_messages);
00314
00315     /*!
00316      * \brief Prints messages of type MSM6 (Full GNSS pseudoranges and phaseranges plus CNR, high
      resolution)
00317      */
00318     std::string print_MSM_6(const Gps_Ephemeris& gps_eph,
00319         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00320         const Galileo_Ephemeris& gal_eph,
00321         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00322         double obs_time,
00323         const std::map<int32_t, Gnss_Synchro>& observables,
00324         uint32_t ref_id,
00325         uint32_t clock_steering_indicator,
00326         uint32_t external_clock_indicator,
00327         int32_t smooth_int,
00328         bool divergence_free,
00329         bool more_messages);
00330
00331     /*!
00332      * \brief Prints messages of type MSM7 (Full GNSS pseudoranges, phaseranges, phaserange rate and
      CNR, high resolution)
00333      */
00334     std::string print_MSM_7(const Gps_Ephemeris& gps_eph,
00335         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00336         const Galileo_Ephemeris& gal_eph,
00337         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00338         double obs_time,
00339         const std::map<int32_t, Gnss_Synchro>& observables,
00340         uint32_t ref_id,
00341         uint32_t clock_steering_indicator,
00342         uint32_t external_clock_indicator,
00343         int32_t smooth_int,
00344         bool divergence_free,
00345         bool more_messages);
00346
00347     /*!
00348      * \brief Prints message type 1057 (SSR GPS Orbit Correction)
00349      */
00350     std::string print_MT1057(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator =
      false);
00351
00352     /*!
00353      * \brief Prints message type 1058 (SSR GPS Clock Correction)
00354      */
00355     std::string print_MT1058(const Galileo_HAS_data& has_data, bool use_clock_subset = false, bool
      ssr_multiple_msg_indicator = false);
00356
00357     /*!
00358      * \brief Prints message type 1059 (SSR GPS Satellite Code Bias)
00359      */
00360     std::string print_MT1059(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator =
      false);
00361
00362     /*!
00363      * \brief Prints message type 1060 (SSR GPS Combined Orbit and Clock Correction)
00364      */
00365     std::string print_MT1060(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator =
      false);
00366
00367     /*!
00368      * \brief Prints messages of type IGM01 (SSR Orbit Correction)
00369      */
00370     std::vector<std::string> print_IGM01(const Galileo_HAS_data& has_data);
00371
00372     /*!
00373      * \brief Prints messages of type IGM02 (SSR Clock Correction)
00374      */
00375     std::vector<std::string> print_IGM02(const Galileo_HAS_data& has_data, bool use_clock_subset =
      false);
00376
00377     /*!
00378      * \brief Prints messages of type IGM03 (SSR Combined Orbit and Clock Correction)
00379      */
00380     std::vector<std::string> print_IGM03(const Galileo_HAS_data& has_data);
00381
00382     /*!
00383      * \brief Prints messages of type IGM05 (SSR Bias Correction)
```

```
00384      */
00385     std::vector<std::string> print_IGM05(const Galileo_HAS_data& has_data);
00386
00387     uint32_t lock_time(const Gps_Ephemeris& eph, double obs_time, const Gnss_Synchro& gnss_synchro);
      //!< Returns the time period in which GPS L1 signals have been continually tracked.
00388     uint32_t lock_time(const Gps_CNAV_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);  //!< Returns the time period in which GPS L2 signals have been continually tracked.
00389     uint32_t lock_time(const Galileo_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);   //!< Returns the time period in which Galileo signals have been continually tracked.
00390
00391     /*!
00392      * \brief Locks time period in which GLONASS signals have been continually tracked.
00393      * \note Code added as part of GSoC 2017 program
00394      * \param eph GLONASS GNAV Broadcast Ephemeris
00395      * \param obs_time Time of observation at the moment of printing
00396      * \param observables Set of observables as defined by the platform
00397      * \return Returns the time period in which GLONASS signals have been continually tracked.
00398      */
00399     uint32_t lock_time(const Glonass_Gnav_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);
00400
00401     std::string bin_to_hex(const std::string& s) const;  //!< Returns a string of hexadecimal symbols
      from a string of binary symbols
00402     std::string hex_to_bin(const std::string& s) const;  //!< Returns a string of binary symbols from
      a string of hexadecimal symbols
00403
00404     std::string bin_to_binary_data(const std::string& s) const;  //!< Returns a string of binary data
      from a string of binary symbols
00405     std::string binary_data_to_bin(const std::string& s) const;  //!< Returns a string of binary
      symbols from a string of binary data
00406
00407     uint32_t bin_to_uint(const std::string& s) const;  //!< Returns an uint32_t from a string of
      binary symbols
00408     int32_t bin_to_int(const std::string& s) const;
00409     double bin_to_double(const std::string& s) const;  //!< Returns double from a string of binary
      symbols
00410     int32_t bin_to_sint(const std::string& s) const;
00411     uint64_t hex_to_uint(const std::string& s) const;  //!< Returns an uint64_t from a string of
      hexadecimal symbols
00412     int64_t hex_to_int(const std::string& s) const;    //!< Returns a int64_t from a string of
      hexadecimal symbols
00413
00414     bool check_CRC(const std::string& message) const;  //!< Checks that the CRC of a RTCM package is
      correct
00415
00416     void run_server();   //!< Starts running the server
00417     void stop_server();  //!< Stops the server
00418
00419     void send_message(const std::string& msg);  //!< Sends a message through the server to all
      connected clients
00420     bool is_server_running() const;             //!< Returns true if the server is running, false
      otherwise
00421
00422 private:
00423     //
00424     // Generation of messages content
00425     //
00426     std::bitset<64> get_MT1001_4_header(uint32_t msg_number,
00427         double obs_time,
00428         const std::map<int32_t, Gnss_Synchro>& observables,
00429         uint32_t ref_id,
00430         uint32_t smooth_int,
00431         bool sync_flag,
00432         bool divergence_free);
00433
00434     std::bitset<58> get_MT1001_sat_content(const Gps_Ephemeris& eph, double obs_time, const
      Gnss_Synchro& gnss_synchro);
00435     std::bitset<74> get_MT1002_sat_content(const Gps_Ephemeris& eph, double obs_time, const
      Gnss_Synchro& gnss_synchro);
00436     std::bitset<101> get_MT1003_sat_content(const Gps_Ephemeris& ephL1, const Gps_CNAV_Ephemeris&
      ephL2, double obs_time, const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
00437     std::bitset<125> get_MT1004_sat_content(const Gps_Ephemeris& ephL1, const Gps_CNAV_Ephemeris&
      ephL2, double obs_time, const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
00438
00439     std::bitset<152> get_MT1005_test();
00440
00441     /*!
00442      * \brief Generates contents of message header for types 1009, 1010, 1011 and 1012. GLONASS RTK
      Message
00443      * \note Code added as part of GSoC 2017 program
00444      * \param msg_number Message type number, acceptable options include 1009 to 1012
00445      * \param obs_time Time of observation at the moment of printing
00446      * \param observables Set of observables as defined by the platform
00447      * \param ref_id
00448      * \param smooth_int
00449      * \param divergence_free
00450      * \return Returns the message header content as set of bits
```

```
00451      */
00452     std::bitset<61> get_MT1009_12_header(uint32_t msg_number,
00453         double obs_time,
00454         const std::map<int32_t, Gnss_Synchro>& observables,
00455         uint32_t ref_id,
00456         uint32_t smooth_int,
00457         bool sync_flag,
00458         bool divergence_free);
00459
00460     /*!
00461      * \brief Get the contents of the satellite specific portion of a type 1009 Message (GLONASS Basic
      RTK, L1 Only)
00462      * \details Contents generated for each satellite. See table 3.5-11
00463      * \note Code added as part of GSoC 2017 program
00464      * \param ephGNAV Ephemeris for GLONASS GNAV in L1 satellites
00465      * \param obs_time Time of observation at the moment of printing
00466      * \param gnss_synchro Information generated by channels while processing the satellite
00467      * \return Returns the message content as set of bits
00468      */
00469     std::bitset<64> get_MT1009_sat_content(const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const Gnss_Synchro& gnss_synchro);
00470     /*!
00471      * \brief Get the contents of the satellite specific portion of a type 1010 Message (GLONASS
      Extended RTK, L1 Only)
00472      * \details Contents generated for each satellite. See table 3.5-12
00473      * \note Code added as part of GSoC 2017 program
00474      * \param ephGNAV Ephemeris for GLONASS GNAV in L1 satellites
00475      * \param obs_time Time of observation at the moment of printing
00476      * \param gnss_synchro Information generated by channels while processing the satellite
00477      * \return Returns the message content as set of bits
00478      */
00479     std::bitset<79> get_MT1010_sat_content(const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const Gnss_Synchro& gnss_synchro);
00480     /*!
00481      * \brief Get the contents of the satellite specific portion of a type 1011 Message (GLONASS Basic
      RTK, L1 & L2)
00482      * \details Contents generated for each satellite. See table 3.5-13
00483      * \note Code added as part of GSoC 2017 program
00484      * \param ephGNAVL1 Ephemeris for GLONASS GNAV in L1 satellites
00485      * \param ephGNAVL2 Ephemeris for GLONASS GNAV in L2 satellites
00486      * \param obs_time Time of observation at the moment of printing
00487      * \param gnss_synchroL1 Information generated by channels while processing the GLONASS GNAV L1
      satellite
00488      * \param gnss_synchroL2 Information generated by channels while processing the GLONASS GNAV L2
      satellite
00489      * \return Returns the message content as set of bits
00490      */
00491     std::bitset<107> get_MT1011_sat_content(const Glonass_Gnav_Ephemeris& ephL1, const
      Glonass_Gnav_Ephemeris& ephL2, double obs_time, const Gnss_Synchro& gnss_synchroL1, const
      Gnss_Synchro& gnss_synchroL2);
00492     /*!
00493      * \brief Get the contents of the satellite specific portion of a type 1012 Message (GLONASS
      Extended RTK, L1 & L2)
00494      * \details Contents generated for each satellite. See table 3.5-14
00495      * \note Code added as part of GSoC 2017 program
00496      * \param ephGNAVL1 Ephemeris for GLONASS GNAV in L1 satellites
00497      * \param ephGNAVL2 Ephemeris for GLONASS GNAV in L2 satellites
00498      * \param obs_time Time of observation at the moment of printing
00499      * \param gnss_synchroL1 Information generated by channels while processing the GLONASS GNAV L1
      satellite
00500      * \param gnss_synchroL2 Information generated by channels while processing the GLONASS GNAV L2
      satellite
00501      * \return Returns the message content as set of bits
00502      */
00503     std::bitset<130> get_MT1012_sat_content(const Glonass_Gnav_Ephemeris& ephL1, const
      Glonass_Gnav_Ephemeris& ephL2, double obs_time, const Gnss_Synchro& gnss_synchroL1, const
      Gnss_Synchro& gnss_synchroL2);
00504
00505     std::string get_MSM_header(uint32_t msg_number,
00506         double obs_time,
00507         const std::map<int32_t, Gnss_Synchro>& observables,
00508         uint32_t ref_id,
00509         uint32_t clock_steering_indicator,
00510         uint32_t external_clock_indicator,
00511         int32_t smooth_int,
00512         bool divergence_free,
00513         bool more_messages);
00514
00515     std::string get_MSM_1_content_sat_data(const std::map<int32_t, Gnss_Synchro>& observables);
00516     std::string get_MSM_4_content_sat_data(const std::map<int32_t, Gnss_Synchro>& observables);
00517     std::string get_MSM_5_content_sat_data(const std::map<int32_t, Gnss_Synchro>& observables);
00518
00519     std::string get_MSM_1_content_signal_data(const std::map<int32_t, Gnss_Synchro>& observables);
00520     std::string get_MSM_2_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00521     std::string get_MSM_3_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
```

```
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00522     std::string get_MSM_4_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00523     std::string get_MSM_5_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00524     std::string get_MSM_6_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00525     std::string get_MSM_7_content_signal_data(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris&
      ephCNAV, const Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time,
      const std::map<int32_t, Gnss_Synchro>& observables);
00526     bool check_MSM_size_limits(uint32_t msg_number, const std::map<int32_t, Gnss_Synchro>&
      observables);
00527
00528     std::string get_MT1057_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00529     std::string get_MT1057_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00530     std::string get_MT1058_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator, bool use_clock_subset = false);
00531     std::string get_MT1058_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index, bool
      use_clock_subset = false);
00532     std::string get_MT1059_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00533     std::string get_MT1059_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00534     std::string get_MT1060_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00535     std::string get_MT1060_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00536
00537     std::string get_IGM01_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00538     std::string get_IGM01_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00539     std::string get_IGM02_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator, bool use_clock_subset = false);
00540     std::string get_IGM02_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index, bool
      use_clock_subset = false);
00541     std::string get_IGM03_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00542     std::string get_IGM03_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00543     std::string get_IGM05_header(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      ssr_multiple_msg_indicator);
00544     std::string get_IGM05_content_sat(const Galileo_HAS_data& has_data, uint8_t nsys_index);
00545
00546     //
00547     // Utilities
00548     //
00549     static bool get_has_data_gps_index(const Galileo_HAS_data& has_data, uint8_t& nsys);
00550     static uint8_t get_MT1057_satellite_count(const Galileo_HAS_data& has_data, uint8_t nsys);
00551     static uint8_t get_MT1059_satellite_count(const Galileo_HAS_data& has_data, uint8_t nsys);
00552     static bool get_MT1059_tracking_mode_id(const std::string& signal, uint8_t& tracking_mode_id);
00553     static uint8_t get_MT1060_satellite_count(const Galileo_HAS_data& has_data, uint8_t nsys);
00554     static bool get_IGM05_tracking_mode_id(uint8_t gnss_id, const std::string& signal, uint8_t&
      tracking_mode_id);
00555     static uint8_t get_IGM02_satellite_count(const Galileo_HAS_data& has_data, uint8_t nsys, bool
      use_clock_subset);
00556     static uint8_t get_IGM05_satellite_count(const Galileo_HAS_data& has_data, uint8_t nsys);
00557     static uint8_t get_iod_ssr(uint8_t has_iod_set_id);
00558     static uint8_t get_gnss_iod_lsb(uint16_t gnss_iod);
00559     static uint32_t get_msm_message_number(char system, uint32_t msm_type);
00560     static uint32_t get_MSM_satellite_data_bits(uint32_t msm_type);
00561     static uint32_t get_MSM_signal_data_bits(uint32_t msm_type);
00562     static uint32_t get_msm_signal_id(const Gnss_Synchro& gnss_synchro);
00563     static std::vector<std::pair<int32_t, Gnss_Synchro» get_ordered_msm_signal_cells(const
      std::map<int32_t, Gnss_Synchro>& observables);
00564     static bool get_msm_signal_wavelength(const Gnss_Synchro& gnss_synchro, double& lambda);
00565     static double get_reconstructed_glonass_l1_pseudorange_m(const Gnss_Synchro& gnss_synchro);
00566     static bool get_msm_glonass_frequency_channel_number(const Gnss_Synchro& gnss_synchro, uint32_t&
      frequency_channel_number);
00567     static std::bitset<4> get_msm_extended_satellite_info(const Gnss_Synchro& gnss_synchro);
00568     static char get_msm_message_system(uint32_t msg_number);
00569     static char get_msm_observable_system(const std::map<int32_t, Gnss_Synchro>& observables);
00570     static char get_msm_ephemeris_system(const Gps_Ephemeris& gps_eph,
00571         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00572         const Galileo_Ephemeris& gal_eph,
00573         const Glonass_Gnav_Ephemeris& glo_gnav_eph);
00574     static uint32_t get_msm_message_number_from_inputs(uint32_t msm_type,
00575         const Gps_Ephemeris& gps_eph,
00576         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00577         const Galileo_Ephemeris& gal_eph,
00578         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00579         const std::map<int32_t, Gnss_Synchro>& observables);
00580     static std::map<std::string, int> galileo_signal_map;
00581     static std::map<std::string, int> gps_signal_map;
00582     std::vector<std::pair<int32_t, Gnss_Synchro» sort_by_signal(const std::vector<std::pair<int32_t,
      Gnss_Synchro»& synchro_map) const;
```

```
00583     std::vector<std::pair<int32_t, Gnss_Synchro» sort_by_PRN_mask(const std::vector<std::pair<int32_t,
      Gnss_Synchro»& synchro_map) const;
00584     boost::posix_time::ptime compute_GPS_time(const Gps_Ephemeris& eph, double obs_time) const;
00585     boost::posix_time::ptime compute_GPS_time(const Gps_CNAV_Ephemeris& eph, double obs_time) const;
00586     boost::posix_time::ptime compute_Galileo_time(const Galileo_Ephemeris& eph, double obs_time)
      const;
00587     boost::posix_time::ptime compute_GLONASS_time(const Glonass_Gnav_Ephemeris& eph, double obs_time)
      const;
00588     boost::posix_time::ptime gps_L1_last_lock_time[64];
00589     boost::posix_time::ptime gps_L2_last_lock_time[64];
00590     boost::posix_time::ptime gal_E1_last_lock_time[64];
00591     boost::posix_time::ptime gal_E5_last_lock_time[64];
00592     boost::posix_time::ptime glo_L1_last_lock_time[64];
00593     boost::posix_time::ptime glo_L2_last_lock_time[64];
00594     uint32_t lock_time_indicator(uint32_t lock_time_period_s);
00595     uint32_t msm_lock_time_indicator(uint32_t lock_time_period_s);
00596     uint32_t msm_extended_lock_time_indicator(uint32_t lock_time_period_s);
00597     static uint32_t clamp_rounded_uint(double value, uint32_t max_value);
00598     uint8_t ssr_update_interval(uint16_t validity_seconds) const;
00599
00600     //
00601     // Classes for TCP communication
00602     //
00603     uint16_t RTCM_port;
00604     // uint16_t RTCM_Station_ID;
00605     class Rtcm_Message
00606     {
00607     public:
00608         static const std::size_t header_length = 6;
00609         static const std::size_t max_body_length = 1029;
00610
00611         Rtcm_Message()
00612             : body_length_(0)
00613         {
00614         }
00615
00616         const char* data() const
00617         {
00618             return data_.data();
00619         }
00620
00621         char* data()
00622         {
00623             return data_.data();
00624         }
00625
00626         inline std::size_t length() const
00627         {
00628             return header_length + body_length_;
00629         }
00630
00631         const char* body() const
00632         {
00633             return data_.data() + header_length;
00634         }
00635
00636         char* body()
00637         {
00638             return data_.data() + header_length;
00639         }
00640
00641         std::size_t body_length() const
00642         {
00643             return body_length_;
00644         }
00645
00646         void body_length(std::size_t new_length)
00647         {
00648             body_length_ = new_length;
00649             if (body_length_ > max_body_length)
00650                 {
00651                     body_length_ = max_body_length;
00652                 }
00653         }
00654
00655         inline bool decode_header()
00656         {
00657             std::string header(data_.data(), header_length);
00658             if (header[0] != 'G' || header[1] != 'S')
00659                 {
00660                     return false;
00661                 }
00662
00663             auto header2 = header.substr(2);
00664             try
00665                 {
00666                     body_length_ = std::stoi(header2);
```

```
00667                 }
00668             catch (const std::exception& e)
00669                 {
00670                     // invalid stoi conversion
00671                     body_length_ = 0;
00672                     return false;
00673                 }
00674 
00675             if (body_length_ == 0)
00676                 {
00677                     return false;
00678                 }
00679 
00680             if (body_length_ > max_body_length)
00681                 {
00682                     body_length_ = 0;
00683                     return false;
00684                 }
00685             return true;
00686         }
00687 
00688         inline void encode_header()
00689         {
00690             std::stringstream ss;
00691             ss « "GS" « std::setw(4) « std::max(std::min(static_cast<int>(body_length_),
      static_cast<int>(max_body_length)), 0);
00692             std::string header = ss.str();
00693             header.resize(header_length, ' ');
00694             std::copy(header.begin(), header.end(), data_.begin());
00695         }
00696 
00697     private:
00698         std::array<char, header_length + max_body_length> data_{};
00699         std::size_t body_length_;
00700     };
00701 
00702 
00703     class RtcmListener
00704     {
00705     public:
00706         virtual ~RtcmListener() = default;
00707         virtual void deliver(const Rtcm_Message& msg) = 0;
00708     };
00709 
00710 
00711     class Rtcm_Listener_Room
00712     {
00713     public:
00714         inline void join(const std::shared_ptr<RtcmListener>& participant)
00715         {
00716             participants_.insert(participant);
00717             for (const auto& msg : recent_msgs_)
00718                 {
00719                     participant->deliver(msg);
00720                 }
00721         }
00722 
00723         inline void leave(const std::shared_ptr<RtcmListener>& participant)
00724         {
00725             participants_.erase(participant);
00726         }
00727 
00728         inline void deliver(const Rtcm_Message& msg)
00729         {
00730             recent_msgs_.push_back(msg);
00731             while (recent_msgs_.size() > max_recent_msgs)
00732                 {
00733                     recent_msgs_.pop_front();
00734                 }
00735 
00736             for (const auto& participant : participants_)
00737                 {
00738                     participant->deliver(msg);
00739                 }
00740         }
00741 
00742     private:
00743         std::set<std::shared_ptr<RtcmListener» participants_;
00744         enum
00745         {
00746             max_recent_msgs = 1
00747         };
00748         std::deque<Rtcm_Message> recent_msgs_;
00749     };
00750 
00751 
00752     class Rtcm_Session
```

```
00753         : public RtcmListener,
00754           public std::enable_shared_from_this<Rtcm_Session>
00755     {
00756     public:
00757         Rtcm_Session(boost::asio::ip::tcp::socket socket, Rtcm_Listener_Room& room) :
      socket_(std::move(socket)), room_(room) {}
00758         inline void start()
00759         {
00760             room_.join(shared_from_this());
00761             do_read_message_header();
00762         }
00763 
00764         inline void deliver(const Rtcm_Message& msg)
00765         {
00766             bool write_in_progress = !write_msgs_.empty();
00767             write_msgs_.push_back(msg);
00768             if (!write_in_progress)
00769                 {
00770                     do_write();
00771                 }
00772         }
00773 
00774     private:
00775         inline void do_read_message_header()
00776         {
00777             auto self(shared_from_this());
00778             boost::asio::async_read(socket_,
00779                 boost::asio::buffer(read_msg_.data(), Rtcm_Message::header_length),
00780                 [this, self](boost::system::error_code ec, std::size_t /*length*/) {
00781                     if (!ec && read_msg_.decode_header())
00782                         {
00783                             do_read_message_body();
00784                         }
00785                     else if (!ec && !read_msg_.decode_header())
00786                         {
00787                             client_says += read_msg_.data();
00788                             bool first = true;
00789                             while (client_says.length() >= 80)
00790                                 {
00791                                     if (first == true)
00792                                         {
00793                                             DLOG(INFO) « "Client says:";
00794                                             first = false;
00795                                         }
00796                                     DLOG(INFO) « client_says;
00797                                     client_says = client_says.substr(80, client_says.length() - 80);
00798                                 }
00799                             do_read_message_header();
00800                         }
00801                     else
00802                         {
00803                             std::cout « "Closing connection with RTCM client\n";
00804                             room_.leave(shared_from_this());
00805                         }
00806                 });
00807         }
00808 
00809         inline void do_read_message_body()
00810         {
00811             auto self(shared_from_this());
00812             boost::asio::async_read(socket_,
00813                 boost::asio::buffer(read_msg_.body(), read_msg_.body_length()),
00814                 [this, self](boost::system::error_code ec, std::size_t /*length*/) {
00815                     if (!ec)
00816                         {
00817                             room_.deliver(read_msg_);
00818                             // std::cout « "Delivered message (session): ";
00819                             // std::cout.write(read_msg_.body(), read_msg_.body_length());
00820                             // std::cout « '\n';
00821                             do_read_message_header();
00822                         }
00823                     else
00824                         {
00825                             std::cout « "Closing connection with RTCM client\n";
00826                             room_.leave(shared_from_this());
00827                         }
00828                 });
00829         }
00830 
00831         inline void do_write()
00832         {
00833             auto self(shared_from_this());
00834             boost::asio::async_write(socket_,
00835                 boost::asio::buffer(write_msgs_.front().body(), write_msgs_.front().body_length()),
00836                 [this, self](boost::system::error_code ec, std::size_t /*length*/) {
00837                     if (!ec)
00838                         {
```

```
00839                             write_msgs_.pop_front();
00840                             if (!write_msgs_.empty())
00841                                 {
00842                                     do_write();
00843                                 }
00844                         }
00845                     else
00846                         {
00847                             std::cout « "Closing connection with RTCM client\n";
00848                             room_.leave(shared_from_this());
00849                         }
00850                 });
00851         }
00852
00853         boost::asio::ip::tcp::socket socket_;
00854         Rtcm_Listener_Room& room_;
00855         Rtcm_Message read_msg_;
00856         std::deque<Rtcm_Message> write_msgs_;
00857         std::string client_says;
00858     };
00859
00860
00861     class Tcp_Internal_Client
00862         : public std::enable_shared_from_this<Tcp_Internal_Client>
00863     {
00864     public:
00865         Tcp_Internal_Client(b_io_context& io_context,
00866 #if BOOST_ASIO_USE_RESOLVER_ITERATOR
00867             boost::asio::ip::tcp::resolver::iterator endpoint_iterator)
00868             : io_context_(io_context), socket_(io_context)
00869         {
00870             do_connect(std::move(endpoint_iterator));
00871         }
00872 #else
00873             boost::asio::ip::tcp::resolver::results_type endpoints)
00874             : io_context_(io_context), socket_(io_context)
00875         {
00876             do_connect(std::move(endpoints));
00877         }
00878 #endif
00879
00880         inline void close()
00881         {
00882 #if BOOST_ASIO_USE_IOCONTEXT_POST
00883             io_context_.post([this]() { socket_.close(); });
00884 #else
00885             boost::asio::post(io_context_, [this]() { socket_.close(); });
00886 #endif
00887         }
00888
00889         inline void write(const Rtcm_Message& msg)
00890         {
00891 #if BOOST_ASIO_USE_IOCONTEXT_POST
00892             io_context_.post(
00893                 [this, &msg]() {
00894 #else
00895             boost::asio::post(io_context_,
00896                 [this, &msg]() {
00897 #endif
00898                     bool write_in_progress = !write_msgs_.empty();
00899                     write_msgs_.push_back(msg);
00900                     if (!write_in_progress)
00901                         {
00902                             do_write();
00903                         }
00904                 });
00905         }
00906
00907     private:
00908 #if BOOST_ASIO_USE_RESOLVER_ITERATOR
00909         inline void do_connect(boost::asio::ip::tcp::resolver::iterator endpoint_iterator)
00910         {
00911             boost::asio::async_connect(socket_, std::move(endpoint_iterator),
00912                 [this](boost::system::error_code ec, boost::asio::ip::tcp::resolver::iterator) {
00913 #else
00914         inline void do_connect(boost::asio::ip::tcp::resolver::results_type endpoints)
00915         {
00916             boost::asio::async_connect(socket_, std::move(endpoints),
00917                 [this](boost::system::error_code ec, boost::asio::ip::tcp::endpoint) {
00918 #endif
00919                     if (!ec)
00920                         {
00921                             do_read_message();
00922                         }
00923                     else
00924                         {
00925                             std::cout « "Server is down.\n";
```

```
00926                         }
00927                 });
00928         }
00929
00930         inline void do_read_message()
00931         {
00932             boost::asio::async_read(socket_,
00933                 boost::asio::buffer(read_msg_.data(), 1029),
00934                 [this](boost::system::error_code ec, std::size_t /*length*/) {
00935                     if (!ec)
00936                         {
00937                             do_read_message();
00938                         }
00939                     else
00940                         {
00941                             std::cout « "Error in client\n";
00942                             socket_.close();
00943                         }
00944                 });
00945         }
00946
00947         inline void do_write()
00948         {
00949             boost::asio::async_write(socket_,
00950                 boost::asio::buffer(write_msgs_.front().data(), write_msgs_.front().length()),
00951                 [this](boost::system::error_code ec, std::size_t /*length*/) {
00952                     if (!ec)
00953                         {
00954                             write_msgs_.pop_front();
00955                             if (!write_msgs_.empty())
00956                                 {
00957                                     do_write();
00958                                 }
00959                         }
00960                     else
00961                         {
00962                             socket_.close();
00963                         }
00964                 });
00965         }
00966
00967         b_io_context& io_context_;
00968         boost::asio::ip::tcp::socket socket_;
00969         Rtcm_Message read_msg_;
00970         std::deque<Rtcm_Message> write_msgs_;
00971     };
00972
00973
00974     class Queue_Reader
00975     {
00976     public:
00977         Queue_Reader(b_io_context& io_context, std::shared_ptr<Concurrent_Queue<std::string»& queue,
    int32_t port) : queue_(queue)
00978         {
00979             boost::asio::ip::tcp::resolver resolver(io_context);
00980             std::string host("localhost");
00981             std::string port_str = std::to_string(port);
00982 #if BOOST_ASIO_USE_RESOLVER_ITERATOR
00983             auto queue_endpoint_iterator = resolver.resolve({host.c_str(), port_str.c_str()});
00984             c = std::make_shared<Tcp_Internal_Client>(io_context, queue_endpoint_iterator);
00985 #else
00986             auto endpoints = resolver.resolve(host, port_str);
00987             c = std::make_shared<Tcp_Internal_Client>(io_context, endpoints);
00988 #endif
00989         }
00990
00991         inline void do_read_queue()
00992         {
00993             for (;;)
00994                 {
00995                     std::string message;
00996                     Rtcm_Message msg;
00997                     queue_->wait_and_pop(message);  // message += '\n';
00998                     if (message == "Goodbye")
00999                         {
01000                             break;
01001                         }
01002
01003                     const char* char_msg = message.c_str();
01004                     msg.body_length(message.length());
01005                     std::copy_n(char_msg, msg.body_length(), msg.body());
01006                     msg.encode_header();
01007                     c->write(msg);
01008                 }
01009         }
01010
01011     private:
```

```
01012         std::shared_ptr<Tcp_Internal_Client> c;
01013         std::shared_ptr<Concurrent_Queue<std::string»& queue_;
01014     };
01015
01016
01017     class Tcp_Server
01018     {
01019     public:
01020         Tcp_Server(b_io_context& io_context, const boost::asio::ip::tcp::endpoint& endpoint)
01021             : acceptor_(io_context), socket_(io_context)
01022         {
01023             acceptor_.open(endpoint.protocol());
01024             acceptor_.set_option(boost::asio::ip::tcp::acceptor::reuse_address(true));
01025             acceptor_.bind(endpoint);
01026             acceptor_.listen();
01027             do_accept();
01028         }
01029
01030         inline void close_server()
01031         {
01032             socket_.close();
01033             acceptor_.close();
01034         }
01035
01036     private:
01037         inline void do_accept()
01038         {
01039             acceptor_.async_accept(socket_, [this](boost::system::error_code ec) {
01040                 if (!ec)
01041                     {
01042                         if (first_client)
01043                             {
01044                                 std::cout « "The TCP/IP server of RTCM messages is up and running.
      Accepting connections ...\n";
01045                                 first_client = false;
01046                             }
01047                         else
01048                             {
01049                                 std::cout « "Starting RTCM TCP/IP server session...\n";
01050                                 boost::system::error_code ec2;
01051                                 boost::asio::ip::tcp::endpoint endpoint =
      socket_.remote_endpoint(ec2);
01052                                 if (ec2)
01053                                     {
01054                                         // Error creating remote_endpoint
01055                                         std::cout « "Error getting remote IP address, closing
      session.\n";
01056                                         LOG(INFO) « "Error getting remote IP address";
01057                                         start_session = false;
01058                                     }
01059                                 else
01060                                     {
01061                                         std::string remote_addr = endpoint.address().to_string();
01062                                         std::cout « "Serving client from " « remote_addr « '\n';
01063                                         LOG(INFO) « "Serving client from " « remote_addr;
01064                                     }
01065                             }
01066                         if (start_session)
01067                             {
01068                                 std::make_shared<Rtcm_Session>(std::move(socket_), room_)->start();
01069                             }
01070                     }
01071                 else
01072                     {
01073                         std::cout « "Error when invoking a RTCM session. " « ec « '\n';
01074                     }
01075                 start_session = true;
01076                 do_accept();
01077             });
01078         }
01079
01080         boost::asio::ip::tcp::acceptor acceptor_;
01081         boost::asio::ip::tcp::socket socket_;
01082         Rtcm_Listener_Room room_;
01083         bool first_client = true;
01084         bool start_session = true;
01085     };
01086
01087     b_io_context io_context;
01088     std::shared_ptr<Concurrent_Queue<std::string» rtcm_message_queue;
01089     std::thread t;
01090     std::thread tq;
01091     std::list<Rtcm::Tcp_Server> servers;
01092     bool server_is_running;
01093     void stop_service();
01094
01095     //
```

```
01096     // Transport Layer
01097     //
01098     std::bitset<8> preamble;
01099     std::bitset<6> reserved_field;
01100     std::string add_CRC(const std::string& m) const;
01101     std::string build_message(const std::string& data) const;  // adds 0s to complete a byte and adds
      the CRC
01102
01103     //
01104     // Data Fields
01105     //
01106     std::bitset<12> DF002;
01107     int32_t set_DF002(uint32_t message_number);
01108
01109     std::bitset<12> DF003;
01110     int32_t set_DF003(uint32_t ref_station_ID);
01111
01112     std::bitset<30> DF004;
01113     int32_t set_DF004(double obs_time);
01114
01115     std::bitset<1> DF005;
01116     int32_t set_DF005(bool sync_flag);
01117
01118     std::bitset<5> DF006;
01119     int32_t set_DF006(const std::map<int32_t, Gnss_Synchro>& observables);
01120
01121     std::bitset<1> DF007;
01122     int32_t set_DF007(bool divergence_free_smoothing_indicator);  // 0 - Divergence-free smoothing not
      used 1 - Divergence-free smoothing used
01123
01124     std::bitset<3> DF008;
01125     int32_t set_DF008(int16_t smoothing_interval);
01126
01127     std::bitset<6> DF009;
01128     int32_t set_DF009(const Gnss_Synchro& gnss_synchro);
01129     int32_t set_DF009(const Gps_Ephemeris& gps_eph);
01130
01131     std::bitset<1> DF010;
01132     int32_t set_DF010(bool code_indicator);
01133
01134     std::bitset<24> DF011;
01135     int32_t set_DF011(const Gnss_Synchro& gnss_synchro);
01136
01137     std::bitset<20> DF012;
01138     int32_t set_DF012(const Gnss_Synchro& gnss_synchro);
01139
01140     std::bitset<7> DF013;
01141     int32_t set_DF013(const Gps_Ephemeris& eph, double obs_time, const Gnss_Synchro& gnss_synchro);
01142
01143     std::bitset<8> DF014;
01144     int32_t set_DF014(const Gnss_Synchro& gnss_synchro);
01145
01146     std::bitset<8> DF015;
01147     int32_t set_DF015(const Gnss_Synchro& gnss_synchro);
01148
01149     std::bitset<14> DF017;
01150     int32_t set_DF017(const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
01151
01152     std::bitset<20> DF018;
01153     int32_t set_DF018(const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
01154
01155     std::bitset<7> DF019;
01156     int32_t set_DF019(const Gps_CNAV_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);
01157
01158     std::bitset<8> DF020;
01159     int32_t set_DF020(const Gnss_Synchro& gnss_synchro);
01160
01161     std::bitset<6> DF021;
01162     int32_t set_DF021();
01163
01164     std::bitset<1> DF022;
01165     int32_t set_DF022(bool gps_indicator);
01166
01167     std::bitset<1> DF023;
01168     int32_t set_DF023(bool glonass_indicator);
01169
01170     std::bitset<1> DF024;
01171     int32_t set_DF024(bool galileo_indicator);
01172
01173     std::bitset<38> DF025;
01174     int32_t set_DF025(double antenna_ECEF_X_m);
01175
01176     std::bitset<38> DF026;
01177     int32_t set_DF026(double antenna_ECEF_Y_m);
01178
01179     std::bitset<38> DF027;
```

```
01180     int32_t set_DF027(double antenna_ECEF_Z_m);
01181
01182     std::bitset<16> DF028;
01183     int32_t set_DF028(double height);
01184
01185     std::bitset<8> DF029;
01186
01187     std::bitset<8> DF031;
01188     int32_t set_DF031(uint32_t antenna_setup_id);
01189
01190     std::bitset<8> DF032;
01191
01192     /*!
01193      * \brief Sets the Data Field value
01194      * \note Code added as part of GSoC 2017 program
01195      * \param obs_time Time of observation at the moment of printing
01196      * \return returns 0 upon success
01197      */
01198     int32_t set_DF034(double obs_time);
01199     std::bitset<27> DF034;  //!< GLONASS Epoch Time (tk)
01200
01201     std::bitset<5> DF035;  //!< No. of GLONASS Satellite Signals Processed
01202     int32_t set_DF035(const std::map<int32_t, Gnss_Synchro>& observables);
01203
01204     std::bitset<1> DF036;  //!< GLONASS Divergence-free Smoothing Indicator
01205     int32_t set_DF036(bool divergence_free_smoothing_indicator);
01206
01207     std::bitset<3> DF037;  //!< GLONASS Smoothing Interval
01208     int32_t set_DF037(int16_t smoothing_interval);
01209
01210     std::bitset<6> DF038;  //!< GLONASS Satellite ID (Satellite Slot Number)
01211     int32_t set_DF038(const Gnss_Synchro& gnss_synchro);
01212     int32_t set_DF038(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01213
01214     std::bitset<1> DF039;  //!< GLONASS L1 Code Indicator
01215     int32_t set_DF039(bool code_indicator);
01216
01217     std::bitset<5> DF040;  //!< GLONASS Satellite Frequency Number
01218     int32_t set_DF040(int32_t frequency_channel_number);
01219     int32_t set_DF040(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01220
01221     std::bitset<25> DF041;  //!< GLONASS L1 Pseudorange
01222     int32_t set_DF041(const Gnss_Synchro& gnss_synchro);
01223
01224     std::bitset<20> DF042;  //!< GLONASS L1 PhaseRange - L1 Pseudorange
01225     int32_t set_DF042(const Gnss_Synchro& gnss_synchro);
01226
01227     std::bitset<7> DF043;  //!< GLONASS L1 Lock Time Indicator
01228     int32_t set_DF043(const Glonass_Gnav_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);
01229
01230     std::bitset<7> DF044;  //!< GLONASS Integer L1 Pseudorange Modulus Ambiguity
01231     int32_t set_DF044(const Gnss_Synchro& gnss_synchro);
01232
01233     std::bitset<8> DF045;  //!< GLONASS L1 CNR
01234     int32_t set_DF045(const Gnss_Synchro& gnss_synchro);
01235
01236     std::bitset<2> DF046;  //!< GLONASS L2 code indicator
01237     int32_t set_DF046(uint16_t code_indicator);
01238
01239     std::bitset<14> DF047;  //!< GLONASS L2 - L1 Pseudorange Difference
01240     int32_t set_DF047(const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
01241
01242     std::bitset<20> DF048;  //!< GLONASS L2 PhaseRange - L1 Pseudorange
01243     int32_t set_DF048(const Gnss_Synchro& gnss_synchroL1, const Gnss_Synchro& gnss_synchroL2);
01244
01245     std::bitset<7> DF049;  //!< GLONASS L2 Lock Time Indicator
01246     int32_t set_DF049(const Glonass_Gnav_Ephemeris& eph, double obs_time, const Gnss_Synchro&
      gnss_synchro);
01247
01248     std::bitset<8> DF050;  //!< GLONASS L2 CNR
01249     int32_t set_DF050(const Gnss_Synchro& gnss_synchro);
01250
01251     std::bitset<16> DF051;
01252     int32_t set_DF051(const Gps_Ephemeris& gps_eph, double obs_time);
01253
01254     std::bitset<17> DF052;
01255     int32_t set_DF052(const Gps_Ephemeris& gps_eph, double obs_time);
01256
01257     // Contents of GPS Satellite Ephemeris Data, Message Type 1019
01258     std::bitset<8> DF071;
01259     int32_t set_DF071(const Gps_Ephemeris& gps_eph);
01260
01261     std::bitset<10> DF076;
01262     int32_t set_DF076(const Gps_Ephemeris& gps_eph);
01263
01264     std::bitset<4> DF077;
```

```
01265     int32_t set_DF077(const Gps_Ephemeris& gps_eph);
01266
01267     std::bitset<2> DF078;
01268     int32_t set_DF078(const Gps_Ephemeris& gps_eph);
01269
01270     std::bitset<14> DF079;
01271     int32_t set_DF079(const Gps_Ephemeris& gps_eph);
01272
01273     std::bitset<8> DF080;
01274     int32_t set_DF080(const Gps_Ephemeris& gps_eph);
01275
01276     std::bitset<16> DF081;
01277     int32_t set_DF081(const Gps_Ephemeris& gps_eph);
01278
01279     std::bitset<8> DF082;
01280     int32_t set_DF082(const Gps_Ephemeris& gps_eph);
01281
01282     std::bitset<16> DF083;
01283     int32_t set_DF083(const Gps_Ephemeris& gps_eph);
01284
01285     std::bitset<22> DF084;
01286     int32_t set_DF084(const Gps_Ephemeris& gps_eph);
01287
01288     std::bitset<10> DF085;
01289     int32_t set_DF085(const Gps_Ephemeris& gps_eph);
01290
01291     std::bitset<16> DF086;
01292     int32_t set_DF086(const Gps_Ephemeris& gps_eph);
01293
01294     std::bitset<16> DF087;
01295     int32_t set_DF087(const Gps_Ephemeris& gps_eph);
01296
01297     std::bitset<32> DF088;
01298     int32_t set_DF088(const Gps_Ephemeris& gps_eph);
01299
01300     std::bitset<16> DF089;
01301     int32_t set_DF089(const Gps_Ephemeris& gps_eph);
01302
01303     std::bitset<32> DF090;
01304     int32_t set_DF090(const Gps_Ephemeris& gps_eph);
01305
01306     std::bitset<16> DF091;
01307     int32_t set_DF091(const Gps_Ephemeris& gps_eph);
01308
01309     std::bitset<32> DF092;
01310     int32_t set_DF092(const Gps_Ephemeris& gps_eph);
01311
01312     std::bitset<16> DF093;
01313     int32_t set_DF093(const Gps_Ephemeris& gps_eph);
01314
01315     std::bitset<16> DF094;
01316     int32_t set_DF094(const Gps_Ephemeris& gps_eph);
01317
01318     std::bitset<32> DF095;
01319     int32_t set_DF095(const Gps_Ephemeris& gps_eph);
01320
01321     std::bitset<16> DF096;
01322     int32_t set_DF096(const Gps_Ephemeris& gps_eph);
01323
01324     std::bitset<32> DF097;
01325     int32_t set_DF097(const Gps_Ephemeris& gps_eph);
01326
01327     std::bitset<16> DF098;
01328     int32_t set_DF098(const Gps_Ephemeris& gps_eph);
01329
01330     std::bitset<32> DF099;
01331     int32_t set_DF099(const Gps_Ephemeris& gps_eph);
01332
01333     std::bitset<24> DF100;
01334     int32_t set_DF100(const Gps_Ephemeris& gps_eph);
01335
01336     std::bitset<8> DF101;
01337     int32_t set_DF101(const Gps_Ephemeris& gps_eph);
01338
01339     std::bitset<6> DF102;
01340     int32_t set_DF102(const Gps_Ephemeris& gps_eph);
01341
01342     std::bitset<1> DF103;
01343     int32_t set_DF103(const Gps_Ephemeris& gps_eph);
01344
01345     std::bitset<1> DF104;  //!< GLONASS Almanac Health
01346     int32_t set_DF104(uint32_t glonass_gnav_alm_health);
01347
01348     std::bitset<1> DF105;  //!< GLONASS Almanac Health Availability Indicator
01349     int32_t set_DF105(uint32_t glonass_gnav_alm_health_ind);
01350
01351     std::bitset<2> DF106;  //!< GLONASS P1 Word
```

```
01352     int32_t set_DF106(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01353
01354     std::bitset<12> DF107;  //!< GLONASS Epoch (tk)
01355     int32_t set_DF107(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01356
01357     std::bitset<1> DF108;  //!< GLONASS MSB of Bn Word
01358     int32_t set_DF108(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01359
01360     std::bitset<1> DF109;  //!< GLONASS P2 Word
01361     int32_t set_DF109(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01362
01363     std::bitset<7> DF110;  //!< GLONASS Ephmeris Epoch (tb)
01364     int32_t set_DF110(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01365
01366     std::bitset<24> DF111;  //!< GLONASS Xn first derivative
01367     int32_t set_DF111(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01368
01369     std::bitset<27> DF112;  //!< GLONASS Xn
01370     int32_t set_DF112(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01371
01372     std::bitset<5> DF113;  //!< GLONASS Xn second derivative
01373     int32_t set_DF113(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01374
01375     std::bitset<24> DF114;  //!< GLONASS Yn first derivative
01376     int32_t set_DF114(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01377
01378     std::bitset<27> DF115;  //!< GLONASS Yn
01379     int32_t set_DF115(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01380
01381     std::bitset<5> DF116;  //!< GLONASS Yn second derivative
01382     int32_t set_DF116(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01383
01384     std::bitset<24> DF117;  //!< GLONASS Zn first derivative
01385     int32_t set_DF117(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01386
01387     std::bitset<27> DF118;  //!< GLONASS Zn
01388     int32_t set_DF118(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01389
01390     std::bitset<5> DF119;  //!< GLONASS Zn second derivative
01391     int32_t set_DF119(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01392
01393     std::bitset<1> DF120;  //!< GLONASS P3
01394     int32_t set_DF120(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01395
01396     std::bitset<11> DF121;  //!< GLONASS GAMMA_N
01397     int32_t set_DF121(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01398
01399     std::bitset<2> DF122;  //!< GLONASS P
01400     int32_t set_DF122(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01401
01402     std::bitset<1> DF123;  //!< GLONASS ln (third string)
01403     int32_t set_DF123(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01404
01405     std::bitset<22> DF124;  //!< GLONASS TAU_N
01406     int32_t set_DF124(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01407
01408     std::bitset<5> DF125;  //!< GLONASS DELTA_TAU_N
01409     int32_t set_DF125(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01410
01411     std::bitset<5> DF126;  //!< GLONASS Eccentricity
01412     int32_t set_DF126(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01413
01414     std::bitset<1> DF127;  //!< GLONASS P4
01415     int32_t set_DF127(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01416
01417     std::bitset<4> DF128;  //!< GLONASS F_T
01418     int32_t set_DF128(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01419
01420     std::bitset<11> DF129;  //!< GLONASS N_T
01421     int32_t set_DF129(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01422
01423     std::bitset<2> DF130;  //!< GLONASS M
01424     int32_t set_DF130(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01425
01426     std::bitset<1> DF131;  //!< GLONASS Availability of additional data
01427     int32_t set_DF131(uint32_t fifth_str_additional_data_ind);
01428
01429     std::bitset<11> DF132;  //!< GLONASS N_A
01430     int32_t set_DF132(const Glonass_Gnav_Utc_Model& glonass_gnav_utc_model);
01431
01432     std::bitset<32> DF133;  //!< GLONASS TAU_C
01433     int32_t set_DF133(const Glonass_Gnav_Utc_Model& glonass_gnav_utc_model);
01434
01435     std::bitset<5> DF134;  //!< GLONASS N_4
01436     int32_t set_DF134(const Glonass_Gnav_Utc_Model& glonass_gnav_utc_model);
01437
01438     std::bitset<22> DF135;  //!< GLONASS TAU_GPS
```

```
01439     int32_t set_DF135(const Glonass_Gnav_Utc_Model& glonass_gnav_utc_model);
01440
01441     std::bitset<1> DF136;  //!< GLONASS L_N (FIFTH STRING)
01442     int32_t set_DF136(const Glonass_Gnav_Ephemeris& glonass_gnav_eph);
01443
01444     std::bitset<1> DF137;
01445     int32_t set_DF137(const Gps_Ephemeris& gps_eph);
01446
01447
01448     std::bitset<1> DF141;
01449     int32_t set_DF141(const Gps_Ephemeris& gps_eph);
01450
01451     std::bitset<1> DF142;
01452     int32_t set_DF142(const Gps_Ephemeris& gps_eph);
01453
01454     std::bitset<30> DF248;
01455     int32_t set_DF248(double obs_time);
01456
01457     // Contents of Galileo F/NAV Satellite Ephemeris Data, Message Type 1045
01458     std::bitset<6> DF252;
01459     int32_t set_DF252(const Galileo_Ephemeris& gal_eph);
01460
01461     std::bitset<12> DF289;
01462     int32_t set_DF289(const Galileo_Ephemeris& gal_eph);
01463
01464     std::bitset<10> DF290;
01465     int32_t set_DF290(const Galileo_Ephemeris& gal_eph);
01466
01467     std::bitset<8> DF291;
01468     int32_t set_DF291(const Galileo_Ephemeris& gal_eph);
01469
01470     std::bitset<14> DF292;
01471     int32_t set_DF292(const Galileo_Ephemeris& gal_eph);
01472
01473     std::bitset<14> DF293;
01474     int32_t set_DF293(const Galileo_Ephemeris& gal_eph);
01475
01476     std::bitset<6> DF294;
01477     int32_t set_DF294(const Galileo_Ephemeris& gal_eph);
01478
01479     std::bitset<21> DF295;
01480     int32_t set_DF295(const Galileo_Ephemeris& gal_eph);
01481
01482     std::bitset<31> DF296;
01483     int32_t set_DF296(const Galileo_Ephemeris& gal_eph);
01484
01485     std::bitset<16> DF297;
01486     int32_t set_DF297(const Galileo_Ephemeris& gal_eph);
01487
01488     std::bitset<16> DF298;
01489     int32_t set_DF298(const Galileo_Ephemeris& gal_eph);
01490
01491     std::bitset<32> DF299;
01492     int32_t set_DF299(const Galileo_Ephemeris& gal_eph);
01493
01494     std::bitset<16> DF300;
01495     int32_t set_DF300(const Galileo_Ephemeris& gal_eph);
01496
01497     std::bitset<32> DF301;
01498     int32_t set_DF301(const Galileo_Ephemeris& gal_eph);
01499
01500     std::bitset<16> DF302;
01501     int32_t set_DF302(const Galileo_Ephemeris& gal_eph);
01502
01503     std::bitset<32> DF303;
01504     int32_t set_DF303(const Galileo_Ephemeris& gal_eph);
01505
01506     std::bitset<14> DF304;
01507     int32_t set_DF304(const Galileo_Ephemeris& gal_eph);
01508
01509     std::bitset<16> DF305;
01510     int32_t set_DF305(const Galileo_Ephemeris& gal_eph);
01511
01512     std::bitset<32> DF306;
01513     int32_t set_DF306(const Galileo_Ephemeris& gal_eph);
01514
01515     std::bitset<16> DF307;
01516     int32_t set_DF307(const Galileo_Ephemeris& gal_eph);
01517
01518     std::bitset<32> DF308;
01519     int32_t set_DF308(const Galileo_Ephemeris& gal_eph);
01520
01521     std::bitset<16> DF309;
01522     int32_t set_DF309(const Galileo_Ephemeris& gal_eph);
01523
01524     std::bitset<32> DF310;
01525     int32_t set_DF310(const Galileo_Ephemeris& gal_eph);
```

```
01526
01527     std::bitset<24> DF311;
01528     int32_t set_DF311(const Galileo_Ephemeris& gal_eph);
01529
01530     std::bitset<10> DF312;
01531     int32_t set_DF312(const Galileo_Ephemeris& gal_eph);
01532
01533     std::bitset<10> DF313;
01534     int32_t set_DF313(const Galileo_Ephemeris& gal_eph);
01535
01536     std::bitset<2> DF314;
01537     int32_t set_DF314(const Galileo_Ephemeris& gal_eph);
01538
01539     std::bitset<1> DF315;
01540     int32_t set_DF315(const Galileo_Ephemeris& gal_eph);
01541
01542     std::bitset<2> DF364;
01543
01544     // Content of message header for MSM1, MSM2, MSM3, MSM4, MSM5, MSM6 and MSM7
01545     std::bitset<1> DF393;
01546     int32_t set_DF393(bool more_messages);  // 1 indicates that more MSMs follow for given physical
      time and reference station ID
01547
01548     std::bitset<64> DF394;
01549     int32_t set_DF394(const std::map<int32_t, Gnss_Synchro>& gnss_synchro);
01550
01551     std::bitset<32> DF395;
01552     int32_t set_DF395(const std::map<int32_t, Gnss_Synchro>& gnss_synchro);
01553
01554     std::string set_DF396(const std::map<int32_t, Gnss_Synchro>& observables);
01555
01556     std::bitset<8> DF397;
01557     int32_t set_DF397(const Gnss_Synchro& gnss_synchro);
01558
01559     std::bitset<10> DF398;
01560     int32_t set_DF398(const Gnss_Synchro& gnss_synchro);
01561
01562     std::bitset<14> DF399;
01563     int32_t set_DF399(const Gnss_Synchro& gnss_synchro);
01564
01565     std::bitset<15> DF400;
01566     int32_t set_DF400(const Gnss_Synchro& gnss_synchro);
01567
01568     std::bitset<22> DF401;
01569     int32_t set_DF401(const Gnss_Synchro& gnss_synchro);
01570
01571     std::bitset<4> DF402;
01572     int32_t set_DF402(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris& ephCNAV, const
      Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time, const
      Gnss_Synchro& gnss_synchro);
01573
01574     std::bitset<6> DF403;
01575     int32_t set_DF403(const Gnss_Synchro& gnss_synchro);
01576
01577     std::bitset<15> DF404;
01578     int32_t set_DF404(const Gnss_Synchro& gnss_synchro);
01579
01580     std::bitset<20> DF405;
01581     int32_t set_DF405(const Gnss_Synchro& gnss_synchro);
01582
01583     std::bitset<24> DF406;
01584     int32_t set_DF406(const Gnss_Synchro& gnss_synchro);
01585
01586     std::bitset<10> DF407;
01587     int32_t set_DF407(const Gps_Ephemeris& ephNAV, const Gps_CNAV_Ephemeris& ephCNAV, const
      Galileo_Ephemeris& ephFNAV, const Glonass_Gnav_Ephemeris& ephGNAV, double obs_time, const
      Gnss_Synchro& gnss_synchro);
01588
01589     std::bitset<10> DF408;
01590     int32_t set_DF408(const Gnss_Synchro& gnss_synchro);
01591
01592     std::bitset<3> DF409;
01593     int32_t set_DF409(uint32_t iods);
01594
01595     std::bitset<2> DF411;
01596     int32_t set_DF411(uint32_t clock_steering_indicator);
01597
01598     std::bitset<2> DF412;
01599     int32_t set_DF412(uint32_t external_clock_indicator);
01600
01601     std::bitset<3> DF416;
01602     int32_t set_DF416(double obs_time);
01603
01604     std::bitset<1> DF417;
01605     int32_t set_DF417(bool using_divergence_free_smoothing);
01606
01607     std::bitset<3> DF418;
```

```
01608     int32_t set_DF418(int32_t carrier_smoothing_interval_s);
01609
01610     std::bitset<1> DF420;
01611     int32_t set_DF420(const Gnss_Synchro& gnss_synchro);
01612
01613     // IGS State Space Representation (SSR) data fields
01614     // see https://files.igs.org/pub/data/format/igs_ssr_v1.pdf
01615     std::bitset<3> IDF001;
01616     void set_IDF001(uint8_t version);
01617
01618     std::bitset<8> IDF002;
01619     void set_IDF002(uint8_t igs_message_number);
01620
01621     std::bitset<20> IDF003;
01622     void set_IDF003(uint32_t tow);
01623
01624     std::bitset<4> IDF004;
01625     void set_IDF004(uint8_t ssr_update_interval);
01626
01627     std::bitset<1> IDF005;
01628     void set_IDF005(bool ssr_multiple_message_indicator);
01629
01630     std::bitset<1> IDF006;
01631     void set_IDF006(bool regional_indicator);
01632
01633     std::bitset<4> IDF007;
01634     void set_IDF007(uint8_t ssr_iod);
01635
01636     std::bitset<16> IDF008;
01637     void set_IDF008(uint16_t ssr_provider_id);
01638
01639     std::bitset<4> IDF009;
01640     void set_IDF009(uint8_t ssr_solution_id);
01641
01642     std::bitset<6> IDF010;
01643     void set_IDF010(uint8_t num_satellites);
01644
01645     std::bitset<6> IDF011;
01646     void set_IDF011(uint8_t gnss_satellite_id);
01647
01648     std::bitset<8> IDF012;
01649     void set_IDF012(uint8_t gnss_iod);
01650
01651     std::bitset<22> IDF013;
01652     void set_IDF013(float delta_orbit_radial_m);
01653
01654     std::bitset<20> IDF014;
01655     void set_IDF014(float delta_orbit_in_track_m);
01656
01657     std::bitset<20> IDF015;
01658     void set_IDF015(float delta_orbit_cross_track_m);
01659
01660     std::bitset<21> IDF016;
01661     void set_IDF016(float dot_orbit_delta_track_m_s);
01662
01663     std::bitset<19> IDF017;
01664     void set_IDF017(float dot_orbit_delta_in_track_m_s);
01665
01666     std::bitset<19> IDF018;
01667     void set_IDF018(float dot_orbit_delta_cross_track_m_s);
01668
01669     std::bitset<22> IDF019;
01670     void set_IDF019(float delta_clock_c0_m);
01671
01672     std::bitset<21> IDF020;
01673     void set_IDF020(float delta_clock_c1_m_s);
01674
01675     std::bitset<27> IDF021;
01676     void set_IDF021(float delta_clock_c2_m_s2);
01677
01678     std::bitset<5> IDF023;
01679     void set_IDF023(uint8_t num_bias_processed);
01680
01681     std::bitset<5> IDF024;
01682     void set_IDF024(uint8_t gnss_signal_tracking_mode_id);
01683
01684     std::bitset<14> IDF025;
01685     void set_IDF025(float code_bias_m);
01686 };
01687
01688
01689 /** \} */
01690 /** \} */
01691 #endif  // GNSS_SDR_RTCM_H
```

13.267 rtcm_printer.h File Reference

Interface of a RTCM 3.2 printer for GNSS-SDR This class provides a implementation of a subset of the RTCM Standard 10403.2 for Differential GNSS Services.

```
#include "signal_enabled_flags.h"
#include <cstdint>
#include <fstream>
#include <map>
#include <memory>
#include <string>
```

Classes

- class Rtcm_Printer

  This class provides a implementation of a subset of the RTCM Standard 10403.2 messages.

13.267.1 Detailed Description

Interface of a RTCM 3.2 printer for GNSS-SDR This class provides a implementation of a subset of the RTCM Standard 10403.2 for Differential GNSS Services.

Author

Carles Fernandez-Prades, 2014. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file rtcm_printer.h.

13.268 rtcm_printer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtcm_printer.h
00003  * \brief Interface of a RTCM 3.2 printer for GNSS-SDR
00004  * This class provides a implementation of a subset of the RTCM Standard 10403.2
00005  * for Differential GNSS Services
00006  *
00007  * \author Carles Fernandez-Prades, 2014. cfernandez(at)cttc.es
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_RTCM_PRINTER_H
00021 #define GNSS_SDR_RTCM_PRINTER_H
00022
00023 #include "signal_enabled_flags.h"
00024 #include <cstdint>  // for int32_t
00025 #include <fstream>  // for std::ofstream
00026 #include <map>      // for std::map
00027 #include <memory>   // std::shared_ptr
00028 #include <string>
00029
00030 /** \addtogroup PVT
00031  * \{ */
00032 /** \addtogroup PVT_libs
00033  * \{ */
00034
00035
00036 class Galileo_Ephemeris;
00037 class Glonass_Gnav_Ephemeris;
00038 class Glonass_Gnav_Utc_Model;
00039 class Gnss_Synchro;
```

```
00040 class Gps_CNAV_Ephemeris;
00041 class Gps_Ephemeris;
00042 class Rtcm;
00043 class Rtklib_Solver;
00044 class Galileo_HAS_data;
00045
00046 /*!
00047  * \brief This class provides a implementation of a subset of the RTCM Standard 10403.2 messages
00048  */
00049 class Rtcm_Printer
00050 {
00051 public:
00052     /*!
00053      * \brief Default constructor.
00054      */
00055     Rtcm_Printer(const std::string& filename,
00056         bool flag_rtcm_file_dump,
00057         bool flag_rtcm_server,
00058         bool flag_rtcm_tty_port,
00059         uint16_t rtcm_tcp_port,
00060         uint16_t rtcm_station_id,
00061         const std::string& rtcm_dump_devname,
00062         uint32_t signal_enabled_flags,
00063         bool time_tag_name = true,
00064         const std::string& base_path = ".");
00065
00066     /*!
00067      * \brief Default destructor.
00068      */
00069     ~Rtcm_Printer() noexcept;
00070
00071     /*!
00072      * \brief Print RTCM messages.
00073      */
00074     void Print_Rtcm_Messages(const Rtklib_Solver* pvt_solver,
00075         const std::map<int, Gnss_Synchro>& gnss_observables_map,
00076         double rx_time,
00077         bool rtcm_MSM_enabled,
00078         bool rtcm_MT1019_enabled,
00079         bool rtcm_MT1020_enabled,
00080         bool rtcm_MT1045_enabled,
00081         bool rtcm_MT1077_enabled,
00082         bool rtcm_MT1087_enabled,
00083         bool rtcm_MT1097_enabled,
00084         bool flag_write_RTCM_MSM_output,
00085         bool flag_write_RTCM_1019_output,
00086         bool flag_write_RTCM_1020_output,
00087         bool flag_write_RTCM_1045_output,
00088         bool enable_rx_clock_correction);
00089
00090     uint32_t lock_time(const Gps_Ephemeris& eph, double obs_time, const Gnss_Synchro& gnss_synchro);
00091     uint32_t lock_time(const Gps_CNAV_Ephemeris& eph, double obs_time, const Gnss_Synchro&
    gnss_synchro);
00092     uint32_t lock_time(const Galileo_Ephemeris& eph, double obs_time, const Gnss_Synchro&
    gnss_synchro);
00093
00094     /*!
00095      * \brief Locks time for logging given GLONASS GNAV Broadcast Ephemeris
00096      * \note Code added as part of GSoC 2017 program
00097      * \params glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00098      * \params obs_time Time of observation at the moment of printing
00099      * \params observables Set of observables as defined by the platform
00100      * \return locked time during logging process
00101      */
00102     uint32_t lock_time(const Glonass_Gnav_Ephemeris& eph, double obs_time, const Gnss_Synchro&
    gnss_synchro);
00103
00104     void Print_IGM_Messages(const Galileo_HAS_data& has_data);
00105     void Print_SSR_Messages(const Galileo_HAS_data& has_data);
00106
00107     std::string print_MT1005_test();  //!<  For testing purposes
00108
00109 private:
00110     bool Print_Rtcm_MT1001(const Gps_Ephemeris& gps_eph, double obs_time, const std::map<int32_t,
    Gnss_Synchro>& observables);
00111     bool Print_Rtcm_MT1002(const Gps_Ephemeris& gps_eph, double obs_time, const std::map<int32_t,
    Gnss_Synchro>& observables);
00112     bool Print_Rtcm_MT1003(const Gps_Ephemeris& gps_eph, const Gps_CNAV_Ephemeris& cnav_eph, double
    obs_time, const std::map<int32_t, Gnss_Synchro>& observables);
00113     bool Print_Rtcm_MT1004(const Gps_Ephemeris& gps_eph, const Gps_CNAV_Ephemeris& cnav_eph, double
    obs_time, const std::map<int32_t, Gnss_Synchro>& observables);
00114
00115     /*
00116      * \brief Prints L1-Only GLONASS RTK Observables
00117      * \details This GLONASS message type is not generally used or supported; type 1012 is to be
    preferred.
00118      * \note Code added as part of GSoC 2017 program
```

```
00119      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00120      * \param obs_time Time of observation at the moment of printing
00121      * \param observables Set of observables as defined by the platform
00122      * \return true or false upon operation success
00123      */
00124     bool Print_Rtcm_MT1009(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, double obs_time, const
      std::map<int32_t, Gnss_Synchro>& observables);
00125
00126     /*
00127      * \brief Prints Extended L1-Only GLONASS RTK Observables
00128      * \details This GLONASS message type is used when only L1 data is present and bandwidth is very
      tight, often 1012 is used in such cases.
00129      * \note Code added as part of GSoC 2017 program
00130      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00131      * \param obs_time Time of observation at the moment of printing
00132      * \param observables Set of observables as defined by the platform
00133      * \return true or false upon operation success
00134      */
00135     bool Print_Rtcm_MT1010(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, double obs_time, const
      std::map<int32_t, Gnss_Synchro>& observables);
00136
00137     /*
00138      * \brief Prints L1&L2 GLONASS RTK Observables
00139      * \details This GLONASS message type is not generally used or supported; type 1012 is to be
      preferred
00140      * \note Code added as part of GSoC 2017 program
00141      * \param glonass_gnav_ephL1 GLONASS L1 GNAV Broadcast Ephemeris for satellite
00142      * \param glonass_gnav_ephL2 GLONASS L2 GNAV Broadcast Ephemeris for satellite
00143      * \param obs_time Time of observation at the moment of printing
00144      * \param observables Set of observables as defined by the platform
00145      * \return true or false upon operation success
00146      */
00147     bool Print_Rtcm_MT1011(const Glonass_Gnav_Ephemeris& glonass_gnav_ephL1, const
      Glonass_Gnav_Ephemeris& glonass_gnav_ephL2, double obs_time, const std::map<int32_t, Gnss_Synchro>&
      observables);
00148
00149     /*
00150      * \brief Prints Extended L1&L2 GLONASS RTK Observables
00151      * \details This GLONASS message type is the most common observational message type, with
      L1/L2/SNR content.  This is one of the most common messages found.
00152      * \note Code added as part of GSoC 2017 program
00153      * \param glonass_gnav_ephL1 GLONASS L1 GNAV Broadcast Ephemeris for satellite
00154      * \param glonass_gnav_ephL2 GLONASS L2 GNAV Broadcast Ephemeris for satellite
00155      * \param obs_time Time of observation at the moment of printing
00156      * \param observables Set of observables as defined by the platform
00157      * \return true or false upon operation success
00158      */
00159     bool Print_Rtcm_MT1012(const Glonass_Gnav_Ephemeris& glonass_gnav_ephL1, const
      Glonass_Gnav_Ephemeris& glonass_gnav_ephL2, double obs_time, const std::map<int32_t, Gnss_Synchro>&
      observables);
00160
00161     bool Print_Rtcm_MT1019(const Gps_Ephemeris& gps_eph);      // GPS Ephemeris, should be broadcast
      in the event that the IODC does not match the IODE, and every 2 minutes.
00162     bool Print_Rtcm_MT1045(const Galileo_Ephemeris& gal_eph);  // Galileo Ephemeris, should be
      broadcast every 2 minutes
00163
00164     /*
00165      * \brief Prints GLONASS GNAV Ephemeris
00166      * \details This GLONASS message should be broadcast every 2 minutes
00167      * \note Code added as part of GSoC 2017 program
00168      * \param glonass_gnav_eph GLONASS GNAV Broadcast Ephemeris
00169      * \param utc_model GLONASS GNAV Clock Information broadcast in string 5
00170      * \return true or false upon operation success
00171      */
00172     bool Print_Rtcm_MT1020(const Glonass_Gnav_Ephemeris& glonass_gnav_eph, const
      Glonass_Gnav_Utc_Model& utc_model);
00173
00174     bool Print_Rtcm_MSM(uint32_t msm_number,
00175         const Gps_Ephemeris& gps_eph,
00176         const Gps_CNAV_Ephemeris& gps_cnav_eph,
00177         const Galileo_Ephemeris& gal_eph,
00178         const Glonass_Gnav_Ephemeris& glo_gnav_eph,
00179         double obs_time,
00180         const std::map<int32_t, Gnss_Synchro>& observables,
00181         uint32_t clock_steering_indicator,
00182         uint32_t external_clock_indicator,
00183         int32_t smooth_int,
00184         bool divergence_free,
00185         bool more_messages);
00186
00187     bool Print_Rtcm_MT1057(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator = false);
      // SSR GPS Orbit Corrections
00188     bool Print_Rtcm_MT1058(const Galileo_HAS_data& has_data, bool use_clock_subset = false, bool
      ssr_multiple_msg_indicator = false);  // SSR GPS Clock Corrections
00189     bool Print_Rtcm_MT1059(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator = false);
      // SSR GPS Code Bias Corrections
00190     bool Print_Rtcm_MT1060(const Galileo_HAS_data& has_data, bool ssr_multiple_msg_indicator = false);
```

```
    // SSR GPS Combined Orbit & Clock Corrections
00191
00192     bool Print_IGM01(const Galileo_HAS_data& has_data);                                  // SSR Orbit
    Corrections
00193     bool Print_IGM02(const Galileo_HAS_data& has_data, bool use_clock_subset = false);  // SSR Clock
    Corrections
00194     bool Print_IGM03(const Galileo_HAS_data& has_data);                                  // SSR
    Combined Orbit & Clock Corrections
00195     bool Print_IGM05(const Galileo_HAS_data& has_data);                                  // SSR Bias
    Corrections
00196
00197     int32_t init_serial(const std::string& serial_device);  // serial port control
00198     void close_serial() const;
00199     bool Print_Message(const std::string& message);
00200
00201     std::unique_ptr<Rtcm> rtcm;
00202     std::ofstream rtcm_file_descriptor;  // Output file stream for RTCM log file
00203     std::string rtcm_filename;           // String with the RTCM log filename
00204     std::string rtcm_base_path;
00205     std::string rtcm_devname;
00206     int32_t rtcm_dev_descriptor;  // RTCM serial device descriptor (i.e. COM port)
00207     uint16_t port;
00208     uint16_t station_id;
00209     bool d_rtcm_has_written_once;
00210     bool d_rtcm_file_dump;
00211     const Signal_Enabled_Flags d_flags;
00212 };
00213
00214
00215 /** \} */
00216 /** \} */
00217 #endif  // GNSS_SDR_RTCM_PRINTER_H
```

13.269 rtklib_solver.h File Reference

PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures.

```
#include "beidou_cnav1_ephemeris.h"
#include "beidou_cnav1_iono.h"
#include "beidou_cnav1_navigation_message.h"
#include "beidou_cnav1_utc_model.h"
#include "beidou_dnav_almanac.h"
#include "beidou_dnav_ephemeris.h"
#include "beidou_dnav_iono.h"
#include "beidou_dnav_utc_model.h"
#include "galileo_almanac.h"
#include "galileo_ephemeris.h"
#include "galileo_ephemeris_store.h"
#include "galileo_has_data.h"
#include "galileo_iono.h"
#include "galileo_reduced_ced.h"
#include "galileo_utc_model.h"
#include "glonass_gnav_almanac.h"
#include "glonass_gnav_ephemeris.h"
#include "glonass_gnav_utc_model.h"
#include "gnss_synchro.h"
#include "gps_almanac.h"
#include "gps_cnav_eop.h"
#include "gps_cnav_ephemeris.h"
#include "gps_cnav_iono.h"
#include "gps_cnav_utc_model.h"
#include "gps_ephemeris.h"
#include "gps_iono.h"
#include "gps_utc_model.h"
#include "monitor_pvt.h"
#include "pvt_conf.h"
#include "pvt_kf.h"
#include "pvt_solution.h"
#include "qzss_cnav_eop.h"
```

```
#include "qzss_cnav_iono.h"
#include "qzss_cnav_utc_model.h"
#include "qzss_iono.h"
#include "qzss_utc_model.h"
#include "rtklib.h"
#include "rtklib_conversions.h"
#include "sbas_raw_message.h"
#include "sbas_rtklib_corrections.h"
#include "sensor_data/sensor_data_aggregator.h"
#include <array>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <map>
#include <string>
#include <utility>
```

Classes

- class Rtklib_Solver

  This class implements a PVT solution based on RTKLIB.

Enumerations

- enum class **Rtklib_Fixed_Base_Status** {
  NOT_REQUESTED , **MISSING_OBSERVATIONS** , **MISSING_POSITION** , **STALE** ,
  INVALID_POSITION , **INSUFFICIENT_COMMON_SATELLITES** , **APPLIED** , **SOLVER_FALLBACK** ,
  SOLVER_FAILURE }

13.269.1 Detailed Description

PVT solver based on rtklib library functions adapted to the GNSS-SDR data flow and structures.

Authors

- 2017, Javier Arribas
- 2017-2026, Carles Fernandez
- 2007-2013, T. Takasu

This is a derived work from RTKLIB http://www.rtklib.com/ The original source code at https://github.com/tomojitakasu/RTKLIB is released under the BSD 2-clause license with an additional exclusive clause that does not apply here. This additional clause is reproduced below:
" The software package includes some companion executive binaries or shared libraries necessary to execute APs on Windows. These licenses succeed to the original ones of these software. "
Neither the executive binaries nor the shared libraries are required by, used or included in GNSS-SDR.

Copyright (C) 2007-2013, T. Takasu Copyright (C) 2017-2019, Javier Arribas Copyright (C) 2017-2026, Carles Fernandez All rights reserved.
SPDX-License-Identifier: BSD-2-Clause

Definition in file rtklib_solver.h.

13.270 rtklib_solver.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtklib_solver.h
00003  * \brief PVT solver based on rtklib library functions adapted to the GNSS-SDR
00004  *  data flow and structures
00005  * \authors <ul>
00006  *          <li> 2017, Javier Arribas
00007  *          <li> 2017-2026, Carles Fernandez
```

```
00008  *          <li> 2007-2013, T. Takasu
00009  *          </ul>
00010  *
00011  * This is a derived work from RTKLIB http://www.rtklib.com/
00012  * The original source code at https://github.com/tomojitakasu/RTKLIB is
00013  * released under the BSD 2-clause license with an additional exclusive clause
00014  * that does not apply here. This additional clause is reproduced below:
00015  *
00016  * " The software package includes some companion executive binaries or shared
00017  * libraries necessary to execute APs on Windows. These licenses succeed to the
00018  * original ones of these software. "
00019  *
00020  * Neither the executive binaries nor the shared libraries are required by, used
00021  * or included in GNSS-SDR.
00022  *
00023  * ----------------------------------------------------------------------------
00024  * Copyright (C) 2007-2013, T. Takasu
00025  * Copyright (C) 2017-2019, Javier Arribas
00026  * Copyright (C) 2017-2026, Carles Fernandez
00027  * All rights reserved.
00028  *
00029  * SPDX-License-Identifier: BSD-2-Clause
00030  *
00031  * ----------------------------------------------------------------------------
00032  */
00033
00034 #ifndef GNSS_SDR_RTKLIB_SOLVER_H
00035 #define GNSS_SDR_RTKLIB_SOLVER_H
00036
00037
00038 #include "beidou_cnav1_ephemeris.h"
00039 #include "beidou_cnav1_iono.h"
00040 #include "beidou_cnav1_navigation_message.h"
00041 #include "beidou_cnav1_utc_model.h"
00042 #include "beidou_dnav_almanac.h"
00043 #include "beidou_dnav_ephemeris.h"
00044 #include "beidou_dnav_iono.h"
00045 #include "beidou_dnav_utc_model.h"
00046 #include "galileo_almanac.h"
00047 #include "galileo_ephemeris.h"
00048 #include "galileo_ephemeris_store.h"
00049 #include "galileo_has_data.h"
00050 #include "galileo_iono.h"
00051 #include "galileo_reduced_ced.h"
00052 #include "galileo_utc_model.h"
00053 #include "glonass_gnav_almanac.h"
00054 #include "glonass_gnav_ephemeris.h"
00055 #include "glonass_gnav_utc_model.h"
00056 #include "gnss_synchro.h"
00057 #include "gps_almanac.h"
00058 #include "gps_cnav_eop.h"
00059 #include "gps_cnav_ephemeris.h"
00060 #include "gps_cnav_iono.h"
00061 #include "gps_cnav_utc_model.h"
00062 #include "gps_ephemeris.h"
00063 #include "gps_iono.h"
00064 #include "gps_utc_model.h"
00065 #include "monitor_pvt.h"
00066 #include "pvt_conf.h"
00067 #include "pvt_kf.h"
00068 #include "pvt_solution.h"
00069 #include "qzss_cnav_eop.h"
00070 #include "qzss_cnav_iono.h"
00071 #include "qzss_cnav_utc_model.h"
00072 #include "qzss_iono.h"
00073 #include "qzss_utc_model.h"
00074 #include "rtklib.h"
00075 #include "rtklib_conversions.h"
00076 #include "sbas_raw_message.h"
00077 #include "sbas_rtklib_corrections.h"
00078 #include "sensor_data/sensor_data_aggregator.h"
00079 #include <array>
00080 #include <cstddef>
00081 #include <cstdint>
00082 #include <fstream>
00083 #include <map>
00084 #include <string>
00085 #include <utility>
00086
00087 /** \addtogroup PVT
00088  * \{ */
00089 /** \addtogroup PVT_libs pvt_libs
00090  * Library for the computation of PVT solutions.
00091  * \{ */
00092
00093 struct Ntrip_Rtcm_Snapshot;
00094
```

```
00095 enum class Rtklib_Fixed_Base_Status
00096 {
00097     NOT_REQUESTED,
00098     MISSING_OBSERVATIONS,
00099     MISSING_POSITION,
00100     STALE,
00101     INVALID_POSITION,
00102     INSUFFICIENT_COMMON_SATELLITES,
00103     APPLIED,
00104     SOLVER_FALLBACK,
00105     SOLVER_FAILURE
00106 };
00107
00108
00109 /*!
00110  * \brief This class implements a PVT solution based on RTKLIB
00111  */
00112 class Rtklib_Solver : public Pvt_Solution
00113 {
00114 public:
00115     //! Minimum satellites common to rover and base (any constellation) for
00116     //! the fixed-base merge; the solver-owned policy the display layer quotes
00117     static constexpr int NTRIP_MIN_COMMON_SATELLITES = 4;
00118
00119     Rtklib_Solver(const rtk_t& rtk,
00120         const Pvt_Conf& conf,
00121         const std::string& dump_filename,
00122         uint32_t signal_enabled_flags,
00123         bool flag_dump_to_file,
00124         bool flag_dump_to_mat);
00125
00126     ~Rtklib_Solver() noexcept override;
00127
00128     Rtklib_Solver(const Rtklib_Solver&) = delete;
00129     Rtklib_Solver& operator=(const Rtklib_Solver&) = delete;
00130     Rtklib_Solver(Rtklib_Solver&&) = delete;
00131     Rtklib_Solver& operator=(Rtklib_Solver&&) = delete;
00132
00133     bool get_PVT(const std::map<int, Gnss_Synchro>& gnss_observables_map,
00134         double kf_update_interval_s,
00135         const SensorDataAggregator& sensor_data_aggregator,
00136         bool dump_this_epoch = true,
00137         const Ntrip_Rtcm_Snapshot* fixed_base = nullptr);
00138
00139     double get_hdop() const override;
00140     double get_vdop() const override;
00141     double get_pdop() const override;
00142     double get_gdop() const override;
00143     Monitor_Pvt get_monitor_pvt() const;
00144     Rtklib_Fixed_Base_Status get_fixed_base_status() const;
00145     double get_fixed_base_age_s() const;
00146     std::size_t get_fixed_base_common_satellites() const;
00147
00148     //! true only if precise ephemeris and clock products are loaded; used to
00149     //! distinguish real PPP from PPP-mode processing with broadcast products
00150     inline bool has_precise_ephemeris() const
00151     {
00152         return d_nav_data.ne > 0 && d_nav_data.nc > 0;
00153     }
00154
00155     //! true if the last call to get_PVT() had what it takes to compute a
00156     //! solution, whether it succeeded or not; false if it returned without
00157     //! trying (fewer than four satellites with ephemeris, or no fixed-base data
00158     //! with the fallback to single-point positioning disabled)
00159     inline bool solution_attempted() const
00160     {
00161         return d_solution_attempted;
00162     }
00163
00164     void store_has_data(const Galileo_HAS_data& new_has_data);
00165     void clear_has_corrections();
00166     void update_has_corrections(const std::map<int, Gnss_Synchro>& obs_map);
00167     void store_sbas_message(const Sbas_Raw_Message& message);
00168     void store_gps_ephemeris(const Gps_Ephemeris& ephemeris);
00169     void clear_gps_ephemerides();
00170     bool store_galileo_ephemeris(const Galileo_Ephemeris& ephemeris);
00171     Galileo_Nav_Message_Type galileo_nav_message_type_for_pvt() const;
00172     bool is_galileo_signal_used_in_pvt(const std::string& signal) const;
00173     bool get_galileo_signal_health(uint32_t prn, const std::string& signal, uint32_t observation_tow,
    bool& healthy) const;
00174
00175     /*!
00176      * \brief Broadcast health of one tracked signal, as reported by the navigation
00177      * message that carries it: GPS/QZSS L1 C/A from the LNAV SV health (almanac
00178      * when no ephemeris has been decoded yet), GPS L2C and QZSS L5 from the CNAV
00179      * per-signal health bits, GPS L5 from the LNAV SV health of the same satellite
00180      * (the CNAV L5 bit is broadcast as unhealthy while GPS L5 is pre-operational),
```

```
00181      * Galileo per signal (see get_galileo_signal_health(),
00182      * the almanac health status of the same signal when no ephemeris is available),
00183      * GLONASS from the GNAV ln/Bn
00184      * flags, BeiDou B1I/B3I from the DNAV SV health and B1C/B2a from the
00185      * B-CNAV1/B-CNAV2 health status. Independent of PVT.use_unhealthy_sats.
00186      * \return true if health information is available for that signal (healthy is then set)
00187      */
00188     bool get_broadcast_signal_health(char system, uint32_t prn, const std::string& signal, uint32_t
      observation_tow, bool& healthy) const;
00189
00190     std::map<int, Galileo_Ephemeris> get_galileo_ephemeris_map_for_pvt() const;
00191     bool select_galileo_ephemeris(uint32_t prn, const std::string& signal, uint32_t observation_tow,
00192         Galileo_Ephemeris& ephemeris, bool& from_reduced_ced) const;
00193
00194     sol_t pvt_sol{};
00195     std::array<ssat_t, MAXSAT> pvt_ssat{};
00196
00197     Galileo_Ephemeris_Store galileo_ephemeris_store;                    //!< Source-aware Galileo
      ephemeris storage
00198     std::map<int, Galileo_Ephemeris> galileo_ephemeris_map;             //!< Compatibility PVT view;
      source-aware storage is authoritative
00199     std::map<int, Galileo_Reduced_CED> galileo_reduced_ced_map;         //!< Map storing provisional
      Galileo Reduced CED
00200     std::map<int, Gps_Ephemeris> gps_ephemeris_map;                     //!< Map storing new
      GPS_Ephemeris
00201     std::map<int, Gps_Ephemeris> gps_previous_ephemeris_map;            //!< Previous GPS/QZSS issue
      retained for SBAS IODE handover
00202     std::map<int, Gps_CNAV_Ephemeris> gps_cnav_ephemeris_map;           //!< Map storing new
      GPS_CNAV_Ephemeris
00203     std::map<int, Glonass_Gnav_Ephemeris> glonass_gnav_ephemeris_map;   //!< Map storing new GLONASS
      GNAV Ephemeris
00204     std::map<int, Beidou_Dnav_Ephemeris> beidou_dnav_ephemeris_map;     //!< Map storing new BeiDou
      DNAV Ephmeris
00205     std::map<int, Beidou_Cnav1_Ephemeris> beidou_cnav1_ephemeris_map;   //!< Map storing BeiDou B-CNAV1
      ephemeris
00206     std::map<int, Beidou_Cnav1_Ephemeris> beidou_cnav2_ephemeris_map;   //!< Map storing BeiDou B-CNAV2
      ephemeris
00207
00208     Galileo_Utc_Model galileo_utc_model;
00209     Galileo_Iono galileo_iono;
00210     std::map<int, Galileo_Almanac> galileo_almanac_map;
00211
00212     Gps_Utc_Model gps_utc_model;
00213     Gps_Iono gps_iono;
00214     std::map<int, Gps_Almanac> gps_almanac_map;
00215
00216     Gps_CNAV_Eop gps_cnav_eop;
00217     Gps_CNAV_Iono gps_cnav_iono;
00218     Gps_CNAV_Utc_Model gps_cnav_utc_model;
00219
00220     Qzss_Utc_Model qzss_utc_model;
00221     Qzss_Iono qzss_iono;
00222
00223     Qzss_CNAV_Eop qzss_cnav_eop;
00224     Qzss_CNAV_Iono qzss_cnav_iono;
00225     Qzss_CNAV_Utc_Model qzss_cnav_utc_model;
00226
00227     Glonass_Gnav_Utc_Model glonass_gnav_utc_model;  //!< Map storing GLONASS GNAV UTC Model
00228     std::map<int, Glonass_Gnav_Almanac> glonass_gnav_almanac_map;
00229     Glonass_Gnav_Almanac glonass_gnav_almanac;  //!< Map storing GLONASS GNAV Almanac Model
00230
00231     Beidou_Dnav_Utc_Model beidou_dnav_utc_model;
00232     Beidou_Dnav_Iono beidou_dnav_iono;
00233     Beidou_Cnav1_Iono beidou_cnav1_iono;
00234     Beidou_Cnav1_Utc_Model beidou_cnav1_utc_model;
00235     std::map<int, Beidou_Dnav_Almanac> beidou_dnav_almanac_map;
00236     std::map<int, Bds3_B1c_PageData> beidou_cnav1_page_data_map;
00237
00238 private:
00239     void update_beidou_observation_wavelengths(const obsd_t& observation);
00240     bool save_matfile() const;
00241     bool prepare_fixed_base_observations(const Ntrip_Rtcm_Snapshot& fixed_base,
00242         int& rover_observation_count,
00243         int& base_observation_count);
00244     int merge_duplicated_rover_observations(int rover_observation_count);
00245     bool galileo_ephemeris_is_usable(const Galileo_Ephemeris& ephemeris, uint32_t observation_tow)
      const;
00246     void update_galileo_observation_wavelengths(const obsd_t& observation);
00247
00248     void check_has_orbit_clock_validity(const std::map<int, Gnss_Synchro>& obs_map);
00249     void get_has_biases(const std::map<int, Gnss_Synchro>& obs_map);
00250     void get_current_has_obs_correction(const std::string& signal, uint32_t tow_obs, int prn, uint8_t
      mask_id, uint8_t iod_set_id);
00251     void clear_applied_has_phase_bias_discontinuity(const HAS_obs_corrections* has_correction, int
      prn);
00252
00253     /*!
```

```
00254      * Substitutes a base-stream broadcast ephemeris for a satellite whose
00255      * navigation message the rover has not decoded yet, inserting both the
00256      * ephemeris and the observation. Owns the per-system week derivation and
00257      * mirrors the rover-decoded path's HAS-correction handling, so the two
00258      * conversion policies cannot drift. Returns false when the base stream
00259      * has no ephemeris for the satellite.
00260      */
00261     bool substitute_base_ephemeris(const Ntrip_Rtcm_Snapshot* fixed_base,
00262         int sat,
00263         const Gnss_Synchro& gnss_synchro,
00264         const std::string& band_key,
00265         std::vector<eph_t>& eph_data,
00266         int& valid_obs,
00267         int glo_valid_obs);
00268     bool has_active_has_do_not_use(const std::string& system, int prn, uint32_t tow_obs) const;
00269     bool has_active_has_orbit_clock(const std::string& system, int prn, uint16_t sis_iod, uint32_t
      tow_obs) const;
00270     bool get_active_has_context(const std::string& system, int prn, uint16_t sis_iod, uint32_t
      tow_obs, uint8_t& mask_id, uint8_t& iod_set_id) const;
00271
00272     void reset_relative_filter();
00273
00274     std::vector<obsd_t> d_obs_data = std::vector<obsd_t>(MAXOBS * 2);
00275     // per-epoch scratch of prepare_fixed_base_observations(); members so
00276     // their capacity is reused across epochs
00277     std::vector<obsd_t> d_fixed_base_rover_scratch;
00278     std::vector<obsd_t> d_fixed_base_base_scratch;
00279     std::array<double, 4> d_dop{};
00280     std::map<int, int> d_rtklib_freq_index;
00281     // NTRIP fixed-base second-band slots, resolved once from the signal
00282     // table and the construction-time channel set (the table scan is not
00283     // for the per-observation path)
00284     int d_ntrip_second_band_slot_gps = 1;
00285     int d_ntrip_second_band_slot_gal = 2;
00286     int d_ntrip_second_band_slot_bds = 2;
00287     std::map<std::string, int> d_rtklib_band_index;
00288
00289     std::map<std::string, std::map<int, HAS_orbit_corrections» d_has_orbit_corrections_store_map;  //
      first key is system, second key is PRN
00290     std::map<std::string, std::map<int, HAS_clock_corrections» d_has_clock_corrections_store_map;  //
      first key is system, second key is PRN
00291     std::map<std::string, std::map<int, uint32_t» d_has_satellite_do_not_use_until;                //
      first key is system, second key is PRN
00292
00293     std::map<std::string, std::map<int, HAS_bias_corrections» d_has_code_bias_store_map;         //
      first key is signal, second key is PRN
00294     std::map<std::string, std::map<int, HAS_bias_corrections» d_has_phase_bias_store_map;        //
      first key is signal, second key is PRN
00295     std::map<std::string, std::map<int, uint8_t» d_has_phase_discontinuity_indicator_store_map;  //
      first key is signal, second key is PRN
00296     std::map<std::string, std::map<int, bool» d_has_phase_bias_discontinuity_map;                //
      first key is signal, second key is PRN
00297
00298     std::map<std::string, std::map<int, HAS_obs_corrections» d_has_obs_corr_map;  // first key is
      signal, second key is PRN
00299
00300     std::string d_dump_filename;
00301     std::ofstream d_dump_file;
00302     rtk_t d_rtk{};
00303     Sbas_Rtklib_Corrections d_sbas_corrections;
00304     nav_t d_nav_data{};
00305     Monitor_Pvt d_monitor_pvt{};
00306     Pvt_Conf d_conf;
00307     Pvt_Kf d_pvt_kf;
00308     uint32_t d_signal_enabled_flags;
00309     std::array<double, 3> d_fixed_base_position_ecef_m{};
00310     gtime_t d_fixed_base_last_observation_time{0, 0.0};
00311     double d_fixed_base_age_s = 0.0;
00312     std::size_t d_fixed_base_common_satellites = 0;
00313     int d_fixed_base_station_id = -1;
00314     Galileo_Nav_Message_Type d_galileo_nav_message_type_for_pvt{Galileo_Nav_Message_Type::INAV};
00315     Rtklib_Fixed_Base_Status d_fixed_base_status = Rtklib_Fixed_Base_Status::NOT_REQUESTED;
00316     bool d_fixed_base_initialized = false;
00317     bool d_fixed_base_was_applied = false;
00318     bool d_duplicated_rover_observations_logged = false;
00319     bool d_solution_attempted = false;
00320     bool d_flag_dump_enabled;
00321     bool d_flag_dump_mat_enabled;
00322 };
00323
00324
00325 /** \} */
00326 /** \} */
00327 #endif  // GNSS_SDR_RTKLIB_SOLVER_H
```

13.271 sbas_rtklib_corrections.h File Reference

Bridge between raw SBAS messages and RTKLIB correction state.

```
#include "rtklib.h"
#include "sbas_raw_message.h"
#include <array>
#include <cstddef>
#include <cstdint>
#include <deque>
```

Classes

- class Sbas_Rtklib_Corrections

  Maintains RTKLIB SBAS state from timestamped raw L1 messages.

13.271.1 Detailed Description

Bridge between raw SBAS messages and RTKLIB correction state.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sbas_rtklib_corrections.h.

13.272 sbas_rtklib_corrections.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sbas_rtklib_corrections.h
00003  * \brief Bridge between raw SBAS messages and RTKLIB correction state
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SBAS_RTKLIB_CORRECTIONS_H
00018 #define GNSS_SDR_SBAS_RTKLIB_CORRECTIONS_H
00019
00020 #include "rtklib.h"
00021 #include "sbas_raw_message.h"
00022 #include <array>
00023 #include <cstddef>
00024 #include <cstdint>
00025 #include <deque>
00026
00027 /** \addtogroup PVT
00028  * \{ */
00029 /** \addtogroup PVT_libs
00030  * \{ */
00031
00032 /*!
00033  * \brief Maintains RTKLIB SBAS state from timestamped raw L1 messages.
00034  *
00035  * Raw messages use the receiver sample-clock time base. update() relates that
00036  * clock to GPST at an observation epoch before passing messages to RTKLIB.
00037  */
00038 class Sbas_Rtklib_Corrections
00039 {
00040 public:
00041     explicit Sbas_Rtklib_Corrections(int32_t selected_prn = 0);
00042
```

```
00043     Sbas_Rtklib_Corrections(const Sbas_Rtklib_Corrections&) = delete;
00044     Sbas_Rtklib_Corrections& operator=(const Sbas_Rtklib_Corrections&) = delete;
00045
00046     void push(const Sbas_Raw_Message& message);
00047     void update(int32_t gps_week, double gps_tow_s, double receiver_sample_stamp_s);
00048     void copy_to(nav_t& navigation_data);
00049
00050     int32_t active_prn() const { return d_active_prn; }
00051     size_t pending_messages() const { return d_pending_messages.size(); }
00052     const sbssat_t& satellite_corrections() const { return d_navigation_data.sbssat; }
00053     const sbsion_t* ionosphere_corrections() const { return d_navigation_data.sbsion; }
00054
00055 private:
00056     static constexpr size_t MAX_PENDING_MESSAGES = 1200U;
00057     static constexpr double FUTURE_MESSAGE_TOLERANCE_S = 0.5;
00058
00059     std::deque<Sbas_Raw_Message> d_pending_messages;
00060     std::array<seph_t, NSATSBS * 2> d_sbas_ephemerides{};
00061     nav_t d_navigation_data{};
00062     int32_t d_selected_prn{};
00063     int32_t d_active_prn{};
00064 };
00065
00066 /** \} */
00067 /** \} */
00068
00069 #endif  // GNSS_SDR_SBAS_RTKLIB_CORRECTIONS_H
```

13.273 secure_string.h File Reference

Self-scrubbing value type for credential-bearing configuration fields.

```
#include "rtklib_secure_wipe.h"
#include <string>
#include <utility>
```

Classes

- class Secure_String

  Value type for credential-bearing configuration fields.

13.273.1 Detailed Description

Self-scrubbing value type for credential-bearing configuration fields.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: BSD-2-Clause

Definition in file secure_string.h.

13.274 secure_string.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file secure_string.h
00003  * \brief Self-scrubbing value type for credential-bearing configuration fields
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: BSD-2-Clause
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
```

```
00017 #ifndef GNSS_SDR_SECURE_STRING_H
00018 #define GNSS_SDR_SECURE_STRING_H
00019
00020 #include "rtklib_secure_wipe.h"
00021 #include <string>
00022 #include <utility>
00023
00024 /** \addtogroup PVT
00025  * \{ */
00026 /** \addtogroup PVT_libs
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief Value type for credential-bearing configuration fields.
00032  *
00033  * Wipes its storage with volatile stores before every release - destruction,
00034  * assignment, clear() - so every copy of a configuration struct that carries
00035  * one scrubs itself, and a new holder of such a copy needs no manual scrub
00036  * site. Assign-once usage is assumed: in-place growth of the underlying
00037  * std::string is not protected.
00038  */
00039 class Secure_String
00040 {
00041 public:
00042     Secure_String() = default;
00043
00044     explicit Secure_String(std::string value) : d_value(std::move(value))
00045     {
00046         // a long secret moves its heap buffer out, but a short one is
00047         // byte-copied under the small-string optimization and its bytes stay
00048         // behind in the moved-from parameter: wipe them before it dies
00049         wipe_residue(value);
00050     }
00051     explicit Secure_String(const char* value) : d_value(value) {}
00052
00053     Secure_String(const Secure_String& other) = default;
00054
00055     Secure_String(Secure_String&& other) noexcept : d_value(std::move(other.d_value))
00056     {
00057         other.clear();
00058     }
00059
00060     Secure_String& operator=(const Secure_String& other)
00061     {
00062         if (this != &other)
00063             {
00064                 clear();
00065                 d_value = other.d_value;
00066             }
00067         return *this;
00068     }
00069
00070     Secure_String& operator=(Secure_String&& other) noexcept
00071     {
00072         if (this != &other)
00073             {
00074                 clear();
00075                 d_value = std::move(other.d_value);
00076                 other.clear();
00077             }
00078         return *this;
00079     }
00080
00081     Secure_String& operator=(std::string value)
00082     {
00083         clear();
00084         d_value = std::move(value);
00085         // the moved-from parameter may keep an SSO copy of the secret
00086         wipe_residue(value);
00087         return *this;
00088     }
00089
00090     ~Secure_String()
00091     {
00092         clear();
00093     }
00094
00095     void clear() noexcept
00096     {
00097         wipe_residue(d_value);
00098     }
00099
00100     const std::string& value() const noexcept
00101     {
00102         return d_value;
00103     }
```

```
00104
00105     bool empty() const noexcept
00106     {
00107         return d_value.empty();
00108     }
00109
00110 private:
00111     // wipes the string's whole allocated buffer, not just its current size:
00112     // a moved-from small string reads empty while its inline buffer still
00113     // holds the secret's bytes, so size()-based wiping would skip exactly
00114     // the residue this class exists to remove. The resize() never
00115     // reallocates (the count is bounded by capacity()) and cannot throw.
00116     static void wipe_residue(std::string& value) noexcept
00117     {
00118         if (value.capacity() > 0)
00119             {
00120                 value.resize(value.capacity(), '\0');
00121                 secure_wipe(&value[0], value.size());
00122             }
00123         value.clear();
00124     }
00125
00126     std::string d_value;
00127 };
00128
00129
00130 /** \} */
00131 /** \} */
00132 #endif  // GNSS_SDR_SECURE_STRING_H
```

13.275 serdes_galileo_eph.h File Reference

Serialization / Deserialization of Galileo_Ephemeris objects using Protocol Buffers.

```
#include "galileo_ephemeris.h"
#include "galileo_ephemeris.pb.h"
#include "protobuf_cleanup_manager.h"
#include <memory>
#include <string>
#include <utility>
```

Classes

- class Serdes_Galileo_Eph

  This class implements serialization and deserialization of Galileo_Ephemeris using Protocol Buffers.

13.275.1 Detailed Description

Serialization / Deserialization of Galileo_Ephemeris objects using Protocol Buffers.

Author

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file serdes_galileo_eph.h.

13.276 serdes_galileo_eph.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file serdes_galileo_eph.h
00003  * \brief Serialization / Deserialization of Galileo_Ephemeris objects using
00004  * Protocol Buffers
00005  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
```

```
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SERDES_GALILEO_EPH_H
00019 #define GNSS_SDR_SERDES_GALILEO_EPH_H
00020
00021 #include "galileo_ephemeris.h"
00022 #include "galileo_ephemeris.pb.h"  // file created by Protocol Buffers at compile time
00023 #include "protobuf_cleanup_manager.h"
00024 #include <memory>
00025 #include <string>
00026 #include <utility>
00027
00028 /** \addtogroup PVT
00029  * \{ */
00030 /** \addtogroup PVT_libs
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This class implements serialization and deserialization of
00036  * Galileo_Ephemeris using Protocol Buffers.
00037  */
00038 class Serdes_Galileo_Eph
00039 {
00040 public:
00041     Serdes_Galileo_Eph()
00042     {
00043         // Verify that the version of the library that we linked against is
00044         // compatible with the version of the headers we compiled against.
00045         GOOGLE_PROTOBUF_VERIFY_VERSION;
00046
00047         // Make sure google::protobuf::ShutdownProtobufLibrary() is called only once,
00048         // at the end of the program execution (see Protobuf_Cleanup_Manager)
00049         Protobuf_Cleanup_Manager::get();
00050     }
00051
00052     inline Serdes_Galileo_Eph(const Serdes_Galileo_Eph& other) noexcept : monitor_(other.monitor_)
      //!< Copy constructor
00053     {
00054     }
00055
00056     inline Serdes_Galileo_Eph& operator=(const Serdes_Galileo_Eph& rhs) noexcept  //!< Copy assignment
      operator
00057     {
00058         Serdes_Galileo_Eph temp(rhs);
00059         std::swap(this->monitor_, temp.monitor_);
00060         return *this;
00061     }
00062
00063     inline Serdes_Galileo_Eph(Serdes_Galileo_Eph&& other) noexcept :
      monitor_(std::move(other.monitor_))  //!< Move constructor
00064     {
00065     }
00066
00067     inline Serdes_Galileo_Eph& operator=(Serdes_Galileo_Eph&& other) noexcept  //!< Move assignment
      operator
00068     {
00069         std::swap(this->monitor_, other.monitor_);
00070         return *this;
00071     }
00072
00073     inline std::string createProtobuffer(const std::shared_ptr<Galileo_Ephemeris> monitor)  //!<
      Serialization into a string
00074     {
00075         monitor_.Clear();
00076
00077         std::string data;
00078
00079         monitor_.set_prn(monitor->PRN);
00080         monitor_.set_m_0(monitor->M_0);
00081         monitor_.set_delta_n(monitor->delta_n);
00082         monitor_.set_ecc(monitor->ecc);
00083         monitor_.set_sqrta(monitor->sqrtA);
00084         monitor_.set_omega_0(monitor->OMEGA_0);
00085         monitor_.set_i_0(monitor->i_0);
00086         monitor_.set_omega(monitor->omega);
00087         monitor_.set_omegadot(monitor->OMEGAdot);
00088         monitor_.set_idot(monitor->idot);
00089         monitor_.set_cuc(monitor->Cuc);
00090         monitor_.set_cus(monitor->Cus);
00091         monitor_.set_crc(monitor->Crc);
00092         monitor_.set_crs(monitor->Crs);
```

```
00093         monitor_.set_cic(monitor->Cic);
00094         monitor_.set_cis(monitor->Cis);
00095         monitor_.set_toe(monitor->toe);
00096         monitor_.set_toc(monitor->toc);
00097         monitor_.set_af0(monitor->af0);
00098         monitor_.set_af1(monitor->af1);
00099         monitor_.set_af2(monitor->af2);
00100         monitor_.set_satclkdrift(monitor->satClkDrift);
00101         monitor_.set_dtr(monitor->dtr);
00102         monitor_.set_wn(monitor->WN);
00103         monitor_.set_tow(monitor->tow);
00104 
00105         // Galileo-specific parameters
00106         monitor_.set_iod_ephemeris(monitor->IOD_ephemeris);
00107         monitor_.set_iod_nav(monitor->IOD_nav);
00108         monitor_.set_sisa(monitor->SISA);
00109         monitor_.set_e5a_hs(monitor->E5a_HS);
00110         monitor_.set_e5b_hs(monitor->E5b_HS);
00111         monitor_.set_e1b_hs(monitor->E1B_HS);
00112         monitor_.set_e5a_dvs(monitor->E5a_DVS);
00113         monitor_.set_e5b_dvs(monitor->E5b_DVS);
00114         monitor_.set_e1b_dvs(monitor->E1B_DVS);
00115         monitor_.set_bgd_e1e5a(monitor->BGD_E1E5a);
00116         monitor_.set_bgd_e1e5b(monitor->BGD_E1E5b);
00117 
    monitor_.set_nav_message_type(static_cast<gnss_sdr::GalileoEphemeris_NavMessageType>(monitor->nav_message_type));
00118 
    monitor_.set_nav_message_source(static_cast<gnss_sdr::GalileoEphemeris_NavMessageSource>(monitor->nav_message_source));
00119 
00120         if (!monitor_.SerializeToString(&data))
00121             {
00122                 return {};
00123             }
00124         return data;
00125     }
00126 
00127     inline Galileo_Ephemeris readProtobuffer(const gnss_sdr::GalileoEphemeris& mon) const  //!<
    Deserialization
00128     {
00129         Galileo_Ephemeris monitor;
00130 
00131         monitor.PRN = mon.prn();
00132         monitor.M_0 = mon.m_0();
00133         monitor.delta_n = mon.delta_n();
00134         monitor.ecc = mon.ecc();
00135         monitor.sqrtA = mon.sqrta();
00136         monitor.OMEGA_0 = mon.omega_0();
00137         monitor.i_0 = mon.i_0();
00138         monitor.omega = mon.omega();
00139         monitor.OMEGAdot = mon.omegadot();
00140         monitor.idot = mon.idot();
00141         monitor.Cuc = mon.cuc();
00142         monitor.Cus = mon.cus();
00143         monitor.Crc = mon.crc();
00144         monitor.Crs = mon.crs();
00145         monitor.Cic = mon.cic();
00146         monitor.Cis = mon.cis();
00147         monitor.toe = mon.toe();
00148         monitor.toc = mon.toc();
00149         monitor.af0 = mon.af0();
00150         monitor.af1 = mon.af1();
00151         monitor.af2 = mon.af2();
00152         monitor.satClkDrift = mon.satclkdrift();
00153         monitor.dtr = mon.dtr();
00154         monitor.WN = mon.wn();
00155         monitor.tow = mon.tow();
00156 
00157         // Galileo-specific parameters
00158         monitor.IOD_ephemeris = mon.iod_ephemeris();
00159         monitor.IOD_nav = mon.iod_nav();
00160         monitor.SISA = mon.sisa();
00161         monitor.E5a_HS = mon.e5a_hs();
00162         monitor.E5b_HS = mon.e5b_hs();
00163         monitor.E1B_HS = mon.e1b_hs();
00164         monitor.E5a_DVS = mon.e5a_dvs();
00165         monitor.E5b_DVS = mon.e5b_dvs();
00166         monitor.E1B_DVS = mon.e1b_dvs();
00167         monitor.BGD_E1E5a = mon.bgd_e1e5a();
00168         monitor.BGD_E1E5b = mon.bgd_e1e5b();
00169         const auto nav_message_type = static_cast<uint32_t>(mon.nav_message_type());
00170         if (nav_message_type <= static_cast<uint32_t>(Galileo_Nav_Message_Type::FNAV))
00171             {
00172                 monitor.nav_message_type = static_cast<Galileo_Nav_Message_Type>(nav_message_type);
00173             }
00174         const auto nav_message_source = static_cast<uint32_t>(mon.nav_message_source());
00175         if (nav_message_source <= static_cast<uint32_t>(Galileo_Nav_Message_Source::E5b))
00176             {
```

```
00177                 monitor.nav_message_source =
      static_cast<Galileo_Nav_Message_Source>(nav_message_source);
00178             }
00179
00180         return monitor;
00181     }
00182
00183 private:
00184     gnss_sdr::GalileoEphemeris monitor_{};
00185 };
00186
00187
00188 /** \} */
00189 /** \} */
00190 #endif  // GGNSS_SDR_SERDES_GALILEO_EPH_H
```

13.277 serdes_gps_eph.h File Reference

Serialization / Deserialization of Gps_Ephemeris objects using Protocol Buffers.

```
#include "gps_ephemeris.h"
#include "gps_ephemeris.pb.h"
#include "protobuf_cleanup_manager.h"
#include <memory>
#include <string>
#include <utility>
```

Classes

- class Serdes_Gps_Eph

  This class implements serialization and deserialization of Gps_Ephemeris objects using Protocol Buffers.

13.277.1 Detailed Description

Serialization / Deserialization of Gps_Ephemeris objects using Protocol Buffers.

Author

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file serdes_gps_eph.h.

13.278 serdes_gps_eph.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file serdes_gps_eph.h
00003  * \brief Serialization / Deserialization of Gps_Ephemeris objects using
00004  * Protocol Buffers
00005  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SERDES_GPS_EPH_H
00019 #define GNSS_SDR_SERDES_GPS_EPH_H
00020
00021 #include "gps_ephemeris.h"
00022 #include "gps_ephemeris.pb.h"  // file created by Protocol Buffers at compile time
00023 #include "protobuf_cleanup_manager.h"
00024 #include <memory>
```

```
00025 #include <string>
00026 #include <utility>
00027
00028 /** \addtogroup PVT
00029  * \{ */
00030 /** \addtogroup PVT_libs
00031  * \{ */
00032
00033 /*!
00034  * \brief This class implements serialization and deserialization of
00035  * Gps_Ephemeris objects using Protocol Buffers.
00036  */
00037 class Serdes_Gps_Eph
00038 {
00039 public:
00040     Serdes_Gps_Eph()
00041     {
00042         // Verify that the version of the library that we linked against is
00043         // compatible with the version of the headers we compiled against.
00044         GOOGLE_PROTOBUF_VERIFY_VERSION;
00045
00046         // Make sure google::protobuf::ShutdownProtobufLibrary() is called only once,
00047         // at the end of the program execution (see Protobuf_Cleanup_Manager)
00048         Protobuf_Cleanup_Manager::get();
00049     }
00050
00051     inline Serdes_Gps_Eph(const Serdes_Gps_Eph& other) noexcept : monitor_(other.monitor_)  //!< Copy
    constructor
00052     {
00053     }
00054
00055     inline Serdes_Gps_Eph& operator=(const Serdes_Gps_Eph& rhs) noexcept  //!< Copy assignment
    operator
00056     {
00057         Serdes_Gps_Eph temp(rhs);
00058         std::swap(this->monitor_, temp.monitor_);
00059         return *this;
00060     }
00061
00062     inline Serdes_Gps_Eph(Serdes_Gps_Eph&& other) noexcept : monitor_(std::move(other.monitor_))  //!<
    Move constructor
00063     {
00064     }
00065
00066     inline Serdes_Gps_Eph& operator=(Serdes_Gps_Eph&& other) noexcept  //!< Move assignment operator
00067     {
00068         std::swap(this->monitor_, other.monitor_);
00069         return *this;
00070     }
00071
00072     inline std::string createProtobuffer(const std::shared_ptr<Gps_Ephemeris> monitor)  //!<
    Serialization into a string
00073     {
00074         monitor_.Clear();
00075         std::string data;
00076
00077         monitor_.set_prn(monitor->PRN);
00078         monitor_.set_m_0(monitor->M_0);
00079         monitor_.set_delta_n(monitor->delta_n);
00080         monitor_.set_ecc(monitor->ecc);
00081         monitor_.set_sqrta(monitor->sqrtA);
00082         monitor_.set_omega_0(monitor->OMEGA_0);
00083         monitor_.set_i_0(monitor->i_0);
00084         monitor_.set_omega(monitor->omega);
00085         monitor_.set_omegadot(monitor->OMEGAdot);
00086         monitor_.set_idot(monitor->idot);
00087         monitor_.set_cuc(monitor->Cuc);
00088         monitor_.set_cus(monitor->Cus);
00089         monitor_.set_crc(monitor->Crc);
00090         monitor_.set_crs(monitor->Crs);
00091         monitor_.set_cic(monitor->Cic);
00092         monitor_.set_cis(monitor->Cis);
00093         monitor_.set_toe(monitor->toe);
00094         monitor_.set_toc(monitor->toc);
00095         monitor_.set_af0(monitor->af0);
00096         monitor_.set_af1(monitor->af1);
00097         monitor_.set_af2(monitor->af2);
00098         monitor_.set_satclkdrift(monitor->satClkDrift);
00099         monitor_.set_dtr(monitor->dtr);
00100         monitor_.set_wn(monitor->WN);
00101         monitor_.set_tow(monitor->tow);
00102
00103         // GPS-specific parameters
00104         monitor_.set_code_on_l2(monitor->code_on_L2);
00105         monitor_.set_l2_p_data_flag(monitor->L2_P_data_flag);
00106         monitor_.set_sv_accuracy(monitor->SV_accuracy);
00107         monitor_.set_sv_health(monitor->SV_health);
```

```
00108         monitor_.set_tgd(monitor->TGD);
00109         monitor_.set_iodc(monitor->IODC);
00110         monitor_.set_iode_sf2(monitor->IODE_SF2);
00111         monitor_.set_iode_sf3(monitor->IODE_SF3);
00112         monitor_.set_aodo(monitor->AODO);
00113         monitor_.set_fit_interval_flag(monitor->fit_interval_flag);
00114         monitor_.set_spare1(monitor->spare1);
00115         monitor_.set_spare2(monitor->spare2);
00116         monitor_.set_integrity_status_flag(monitor->integrity_status_flag);
00117         monitor_.set_alert_flag(monitor->alert_flag);
00118         monitor_.set_antispoofing_flag(monitor->antispoofing_flag);
00119 
00120         if (!monitor_.SerializeToString(&data))
00121             {
00122                 return {};
00123             }
00124         return data;
00125     }
00126 
00127     inline Gps_Ephemeris readProtobuffer(const gnss_sdr::GpsEphemeris& mon) const  //!<
      Deserialization
00128     {
00129         Gps_Ephemeris monitor;
00130 
00131         monitor.PRN = mon.prn();
00132         monitor.M_0 = mon.m_0();
00133         monitor.delta_n = mon.delta_n();
00134         monitor.ecc = mon.ecc();
00135         monitor.sqrtA = mon.sqrta();
00136         monitor.OMEGA_0 = mon.omega_0();
00137         monitor.i_0 = mon.i_0();
00138         monitor.omega = mon.omega();
00139         monitor.OMEGAdot = mon.omegadot();
00140         monitor.idot = mon.idot();
00141         monitor.Cuc = mon.cuc();
00142         monitor.Cus = mon.cus();
00143         monitor.Crc = mon.crc();
00144         monitor.Crs = mon.crs();
00145         monitor.Cic = mon.cic();
00146         monitor.Cis = mon.cis();
00147         monitor.toe = mon.toe();
00148         monitor.toc = mon.toc();
00149         monitor.af0 = mon.af0();
00150         monitor.af1 = mon.af1();
00151         monitor.af2 = mon.af2();
00152         monitor.satClkDrift = mon.satclkdrift();
00153         monitor.dtr = mon.dtr();
00154         monitor.WN = mon.wn();
00155         monitor.tow = mon.tow();
00156 
00157         // GPS-specific parameters
00158         monitor.code_on_L2 = mon.code_on_l2();
00159         monitor.L2_P_data_flag = mon.l2_p_data_flag();
00160         monitor.SV_accuracy = mon.sv_accuracy();
00161         monitor.SV_health = mon.sv_health();
00162         monitor.TGD = mon.tgd();
00163         monitor.IODC = mon.iodc();
00164         monitor.IODE_SF2 = mon.iode_sf2();
00165         monitor.IODE_SF3 = mon.iode_sf3();
00166         monitor.AODO = mon.aodo();
00167         monitor.fit_interval_flag = mon.fit_interval_flag();
00168         monitor.spare1 = mon.spare1();
00169         monitor.spare2 = mon.spare2();
00170         monitor.integrity_status_flag = mon.integrity_status_flag();
00171         monitor.alert_flag = mon.alert_flag();
00172         monitor.antispoofing_flag = mon.antispoofing_flag();
00173 
00174         return monitor;
00175     }
00176 
00177 private:
00178     gnss_sdr::GpsEphemeris monitor_{};
00179 };
00180 
00181 
00182 /** \} */
00183 /** \} */
00184 #endif  // GNSS_SDR_SERDES_GPS_EPH_H
```

13.279 serdes_monitor_pvt.h File Reference

Serialization / Deserialization of Monitor_Pvt objects using Protocol Buffers.

```
#include "monitor_pvt.h"
#include "monitor_pvt.pb.h"
#include "protobuf_cleanup_manager.h"
#include <memory>
#include <string>
#include <utility>
```

Classes

- class Serdes_Monitor_Pvt

  This class implements serialization and deserialization of Monitor_Pvt objects using Protocol Buffers.

13.279.1 Detailed Description

Serialization / Deserialization of Monitor_Pvt objects using Protocol Buffers.

Author

Carles Fernandez-Prades, 2019. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file serdes_monitor_pvt.h.

13.280 serdes_monitor_pvt.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file serdes_monitor_pvt.h
00003  * \brief Serialization / Deserialization of Monitor_Pvt objects using
00004  * Protocol Buffers
00005  * \author Carles Fernandez-Prades, 2019. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SERDES_MONITOR_PVT_H
00019 #define GNSS_SDR_SERDES_MONITOR_PVT_H
00020
00021 #include "monitor_pvt.h"
00022 #include "monitor_pvt.pb.h"  // file created by Protocol Buffers at compile time
00023 #include "protobuf_cleanup_manager.h"
00024 #include <memory>
00025 #include <string>
00026 #include <utility>
00027
00028 /** \addtogroup PVT
00029  * \{ */
00030 /** \addtogroup PVT_libs
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This class implements serialization and deserialization of
00036  * Monitor_Pvt objects using Protocol Buffers.
00037  */
00038 class Serdes_Monitor_Pvt
00039 {
00040 public:
00041     Serdes_Monitor_Pvt()
00042     {
00043         // Verify that the version of the library that we linked against is
00044         // compatible with the version of the headers we compiled against.
00045         GOOGLE_PROTOBUF_VERIFY_VERSION;
00046
```

```
00047         // Make sure google::protobuf::ShutdownProtobufLibrary() is called only once,
00048         // at the end of the program execution (see Protobuf_Cleanup_Manager)
00049         Protobuf_Cleanup_Manager::get();
00050     }
00051
00052     inline Serdes_Monitor_Pvt(const Serdes_Monitor_Pvt& other) noexcept : monitor_(other.monitor_)
    //!< Copy constructor
00053     {
00054     }
00055
00056     inline Serdes_Monitor_Pvt& operator=(const Serdes_Monitor_Pvt& rhs) noexcept  //!< Copy assignment
    operator
00057     {
00058         if (this != &rhs)
00059             {
00060                 this->monitor_.CopyFrom(rhs.monitor_);
00061             }
00062         return *this;
00063     }
00064
00065     inline Serdes_Monitor_Pvt(Serdes_Monitor_Pvt&& other) noexcept :
    monitor_(std::move(other.monitor_))  //!< Move constructor
00066     {
00067         // Set the other object's monitor_ to a default-constructed state
00068         other.monitor_ = gnss_sdr::MonitorPvt{};
00069     }
00070
00071     inline Serdes_Monitor_Pvt& operator=(Serdes_Monitor_Pvt&& other) noexcept  //!< Move assignment
    operator
00072     {
00073         if (this != &other)
00074             {
00075                 this->monitor_ = std::move(other.monitor_);
00076             }
00077         return *this;
00078     }
00079
00080     inline std::string createProtobuffer(const Monitor_Pvt* const monitor)  //!< Serialization into a
    string
00081     {
00082         monitor_.Clear();
00083
00084         std::string data;
00085
00086         monitor_.set_tow_at_current_symbol_ms(monitor->TOW_at_current_symbol_ms);
00087         monitor_.set_week(monitor->week);
00088         monitor_.set_rx_time(monitor->RX_time);
00089         monitor_.set_user_clk_offset(monitor->user_clk_offset);
00090         monitor_.set_pos_x(monitor->pos_x);
00091         monitor_.set_pos_y(monitor->pos_y);
00092         monitor_.set_pos_z(monitor->pos_z);
00093         monitor_.set_vel_x(monitor->vel_x);
00094         monitor_.set_vel_y(monitor->vel_y);
00095         monitor_.set_vel_z(monitor->vel_z);
00096         monitor_.set_cov_xx(monitor->cov_xx);
00097         monitor_.set_cov_yy(monitor->cov_yy);
00098         monitor_.set_cov_zz(monitor->cov_zz);
00099         monitor_.set_cov_xy(monitor->cov_xy);
00100         monitor_.set_cov_yz(monitor->cov_yz);
00101         monitor_.set_cov_zx(monitor->cov_zx);
00102         monitor_.set_latitude(monitor->latitude);
00103         monitor_.set_longitude(monitor->longitude);
00104         monitor_.set_height(monitor->height);
00105         monitor_.set_valid_sats(monitor->valid_sats);
00106         monitor_.set_solution_status(monitor->solution_status);
00107         monitor_.set_solution_type(monitor->solution_type);
00108         monitor_.set_ar_ratio_factor(monitor->AR_ratio_factor);
00109         monitor_.set_ar_ratio_threshold(monitor->AR_ratio_threshold);
00110         monitor_.set_gdop(monitor->gdop);
00111         monitor_.set_pdop(monitor->pdop);
00112         monitor_.set_hdop(monitor->hdop);
00113         monitor_.set_vdop(monitor->vdop);
00114         monitor_.set_user_clk_drift_ppm(monitor->user_clk_drift_ppm);
00115         monitor_.set_utc_time(monitor->utc_time);
00116         monitor_.set_vel_e(monitor->vel_e);
00117         monitor_.set_vel_n(monitor->vel_n);
00118         monitor_.set_vel_u(monitor->vel_u);
00119         monitor_.set_cog(monitor->cog);
00120         monitor_.set_galhas_status(monitor->galhas_status);
00121         monitor_.set_geohash(monitor->geohash);
00122
00123         for (const auto& sat : monitor->tracked_satellites)
00124             {
00125                 gnss_sdr::MonitorPvt::TrackedSatellite* pb_sat = monitor_.add_tracked_satellites();
00126                 pb_sat->set_prn(sat.prn);
00127                 pb_sat->set_system(std::string(1, sat.system));
00128                 pb_sat->set_signal(sat.signal);
```

```
00129                 pb_sat->set_azimuth_deg(sat.azimuth_deg);
00130                 pb_sat->set_elevation_deg(sat.elevation_deg);
00131                 pb_sat->set_combined(sat.combined);
00132                 pb_sat->set_used(sat.used);
00133                 pb_sat->set_healthy(sat.healthy);
00134             }
00135
00136         if (!monitor_.SerializeToString(&data))
00137             {
00138                 return {};
00139             }
00140         return data;
00141     }
00142
00143     inline Monitor_Pvt readProtobuffer(const gnss_sdr::MonitorPvt& mon) const  //!< Deserialization
00144     {
00145         Monitor_Pvt monitor;
00146
00147         monitor.TOW_at_current_symbol_ms = mon.tow_at_current_symbol_ms();
00148         monitor.week = mon.week();
00149         monitor.RX_time = mon.rx_time();
00150         monitor.user_clk_offset = mon.user_clk_offset();
00151         monitor.pos_x = mon.pos_x();
00152         monitor.pos_y = mon.pos_y();
00153         monitor.pos_z = mon.pos_z();
00154         monitor.vel_x = mon.vel_x();
00155         monitor.vel_y = mon.vel_y();
00156         monitor.vel_z = mon.vel_z();
00157         monitor.cov_xx = mon.cov_xx();
00158         monitor.cov_yy = mon.cov_yy();
00159         monitor.cov_zz = mon.cov_zz();
00160         monitor.cov_xy = mon.cov_xy();
00161         monitor.cov_yz = mon.cov_yz();
00162         monitor.cov_zx = mon.cov_zx();
00163         monitor.latitude = mon.latitude();
00164         monitor.longitude = mon.longitude();
00165         monitor.height = mon.height();
00166         monitor.valid_sats = static_cast<uint8_t>(mon.valid_sats());
00167         monitor.solution_status = static_cast<uint8_t>(mon.solution_status());
00168         monitor.solution_type = static_cast<uint8_t>(mon.solution_type());
00169         monitor.AR_ratio_factor = mon.ar_ratio_factor();
00170         monitor.AR_ratio_threshold = mon.ar_ratio_threshold();
00171         monitor.gdop = mon.gdop();
00172         monitor.pdop = mon.pdop();
00173         monitor.hdop = mon.hdop();
00174         monitor.vdop = mon.vdop();
00175         monitor.user_clk_drift_ppm = mon.user_clk_drift_ppm();
00176         monitor.utc_time = mon.utc_time();
00177         monitor.vel_e = mon.vel_e();
00178         monitor.vel_n = mon.vel_n();
00179         monitor.vel_u = mon.vel_u();
00180         monitor.cog = mon.cog();
00181         monitor.galhas_status = mon.galhas_status();
00182         monitor.geohash = mon.geohash();
00183
00184         for (const auto& pb_sat : mon.tracked_satellites())
00185             {
00186                 Monitor_Pvt::TrackedSatelliteInfo sat;
00187                 sat.prn = pb_sat.prn();
00188                 sat.system = pb_sat.system().empty() ? '\0' : pb_sat.system()[0];
00189                 sat.signal = pb_sat.signal();
00190                 sat.azimuth_deg = pb_sat.azimuth_deg();
00191                 sat.elevation_deg = pb_sat.elevation_deg();
00192                 sat.combined = pb_sat.combined();
00193                 sat.used = pb_sat.used();
00194                 sat.healthy = pb_sat.healthy();
00195                 monitor.tracked_satellites.push_back(sat);
00196             }
00197
00198         return monitor;
00199     }
00200
00201 private:
00202     gnss_sdr::MonitorPvt monitor_{};
00203 };
00204
00205
00206 /** \} */
00207 /** \} */
00208 #endif  // GNSS_SDR_SERDES_MONITOR_PVT_H
```

13.281 signal_enabled_flags.h File Reference

Class to check the enabled signals.

```
#include "signal_flag.h"
#include <cstdint>
```

Classes

- class Signal_Enabled_Flags

13.281.1 Detailed Description

Class to check the enabled signals.

Author

Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file signal_enabled_flags.h.

13.282 signal_enabled_flags.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_enabled_flags.h
00003  * \brief Class to check the enabled signals
00004  * \author Mathieu Favreau, 2025. favreau.mathieu(at)hotmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SIGNAL_ENABLED_FLAGS_H
00018 #define GNSS_SDR_SIGNAL_ENABLED_FLAGS_H
00019
00020 #include "signal_flag.h"
00021 #include <cstdint>
00022
00023 class ConfigurationInterface;
00024
00025 class Signal_Enabled_Flags
00026 {
00027 public:
00028     explicit Signal_Enabled_Flags(const ConfigurationInterface* configuration);
00029     explicit Signal_Enabled_Flags(uint32_t flags_);
00030
00031     template <typename T>
00032     uint32_t or_all(const T& value) const
00033     {
00034         return value;
00035     }
00036
00037     template <typename T, typename... Args>
00038     uint32_t or_all(const T& first, const Args&... rest) const
00039     {
00040         return first | or_all(rest...);
00041     }
00042
00043     template <typename... Args>
00044     bool check_only_enabled(const Args&... args) const
00045     {
00046         return (flags ^ or_all(args...)) == 0;
00047     }
00048
00049     template <typename... Args>
00050     bool check_any_enabled(const Args&... args) const
00051     {
00052         return (flags & or_all(args...)) > 0;
00053     }
00054
```

```
00055     const uint32_t flags;
00056
00057     const bool has_gps;
00058     const bool has_galileo;
00059     const bool has_glonass;
00060     const bool has_beidou;
00061     const bool has_qzss;
00062
00063     const bool only_gps;
00064     const bool only_galileo;
00065     const bool only_glonass;
00066     const bool only_beidou;
00067     const bool only_qzss;
00068 };
00069
00070 #endif  // GNSS_SDR_SIGNAL_ENABLED_FLAGS_H
```

13.283 direct_resampler_conditioner.h File Reference

Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.

```
#include "gnss_block_interface.h"
#include <gnuradio/hier_block2.h>
#include <string>
```

Classes

- class DirectResamplerConditioner

  Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.

13.283.1 Detailed Description

Interface of an adapter of a direct resampler conditioner block to a SignalConditionerInterface.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file direct_resampler_conditioner.h.

13.284 direct_resampler_conditioner.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file direct_resampler_conditioner.h
00003  * \brief Interface of an adapter of a direct resampler conditioner block
00004  * to a SignalConditionerInterface
00005  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_H
00020 #define GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/hier_block2.h>
00024 #include <string>
00025
00026 /** \addtogroup Resampler
00027  * \{ */
```

```
00028 /** \addtogroup Resampler_adapters
00029  * \{ */
00030
00031
00032 class ConfigurationInterface;
00033
00034 /*!
00035  * \brief Interface of an adapter of a direct resampler conditioner block
00036  * to a SignalConditionerInterface
00037  */
00038 class DirectResamplerConditioner : public GNSSBlockInterface
00039 {
00040 public:
00041     DirectResamplerConditioner(const ConfigurationInterface* configuration,
00042         const std::string& role, unsigned int in_stream,
00043         unsigned int out_stream);
00044
00045     ~DirectResamplerConditioner() = default;
00046
00047     inline std::string role() override
00048     {
00049         return role_;
00050     }
00051
00052     //! Returns "Direct_Resampler"
00053     inline std::string implementation() override
00054     {
00055         return "Direct_Resampler";
00056     }
00057
00058     inline size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block() override;
00067
00068 private:
00069     gr::block_sptr resampler_;
00070     gr::block_sptr file_sink_;
00071     std::string role_;
00072     std::string item_type_;
00073     std::string dump_filename_;
00074     double sample_freq_in_;
00075     double sample_freq_out_;
00076     size_t item_size_;
00077     unsigned int in_stream_;
00078     unsigned int out_stream_;
00079     bool dump_;
00080 };
00081
00082
00083 /** \} */
00084 /** \} */
00085 #endif  // GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_H
```

13.285 mmse_resampler_conditioner.h File Reference

Interface of an adapter of a mmse resampler conditioner block to a SignalConditionerInterface.

```
#include "gnss_block_interface.h"
#include <gnuradio/filter/fir_filter_ccf.h>
#include <gnuradio/filter/fractional_resampler_cc.h>
#include <gnuradio/filter/firdes.h>
#include <string>
```

Classes

- class MmseResamplerConditioner

  Interface of a MMSE resampler block adapter to a SignalConditionerInterface.

13.285.1 Detailed Description

Interface of an adapter of a mmse resampler conditioner block to a SignalConditionerInterface.

Author

Antonio Ramos, 2018. antonio.ramos(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file mmse_resampler_conditioner.h.

13.286 mmse_resampler_conditioner.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file mmse_resampler_conditioner.h
00003  * \brief Interface of an adapter of a mmse resampler conditioner block
00004  * to a SignalConditionerInterface
00005  * \author Antonio Ramos, 2018. antonio.ramos(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_MMSE_RESAMPLER_CONDITIONER_H
00020 #define GNSS_SDR_MMSE_RESAMPLER_CONDITIONER_H
00021
00022 #include "gnss_block_interface.h"
00023 #ifdef GR_GREATER_38
00024 #include <gnuradio/filter/fir_filter_blk.h>
00025 #include <gnuradio/filter/mmse_resampler_cc.h>
00026 #else
00027 #include <gnuradio/filter/fir_filter_ccf.h>
00028 #include <gnuradio/filter/fractional_resampler_cc.h>
00029 #endif
00030
00031 #include <gnuradio/filter/firdes.h>
00032 #include <string>
00033
00034 /** \addtogroup Resampler
00035  * Classes for input signal resampling
00036  * \{ */
00037 /** \addtogroup Resampler_adapters resampler_adapters
00038  * Classes that wrap GNU Radio resampler blocks with a GNSSBlockInterface
00039  * \{ */
00040
00041
00042 class ConfigurationInterface;
00043
00044 /*!
00045  * \brief Interface of a MMSE resampler block adapter
00046  * to a SignalConditionerInterface
00047  */
00048 class MmseResamplerConditioner : public GNSSBlockInterface
00049 {
00050 public:
00051     MmseResamplerConditioner(const ConfigurationInterface* configuration,
00052         const std::string& role, unsigned int in_stream,
00053         unsigned int out_stream);
00054
00055     ~MmseResamplerConditioner() = default;
00056
00057     inline std::string role() override
00058     {
00059         return role_;
00060     }
00061
00062     inline std::string implementation() override
00063     {
00064         return "Mmse_Resampler";
00065     }
00066
00067     inline size_t item_size() override
00068     {
00069         return item_size_;
00070     }
00071
00072     void connect(gr::top_block_sptr top_block) override;
```

```
00073     void disconnect(gr::top_block_sptr top_block) override;
00074     gr::basic_block_sptr get_left_block() override;
00075     gr::basic_block_sptr get_right_block() override;
00076
00077 private:
00078 #ifdef GR_GREATER_38
00079     gr::filter::mmse_resampler_cc::sptr resampler_;
00080 #else
00081     gr::filter::fractional_resampler_cc::sptr resampler_;
00082 #endif
00083     gr::filter::fir_filter_ccf::sptr fir_filter_ccf_;
00084     gr::block_sptr file_sink_;
00085     std::string role_;
00086     std::string item_type_;
00087     std::string dump_filename_;
00088     size_t item_size_;
00089     double sample_freq_in_;
00090     double sample_freq_out_;
00091     unsigned int in_stream_;
00092     unsigned int out_stream_;
00093     bool dump_;
00094 };
00095
00096
00097 /** \} */
00098 /** \} */
00099 #endif  // GNSS_SDR_FRACTIONAL_RESAMPLER_CONDITIONER_H
```

13.287 direct_resampler_conditioner_cb.h File Reference

Nearest neighborhood resampler with std::complex<signed char> input and std::complex<signed char> output.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <cstdint>
```

Classes

- class direct_resampler_conditioner_cb

  This class implements a direct resampler conditioner for std::complex<signed char>.

Typedefs

- using direct_resampler_conditioner_cb_sptr = gnss_shared_ptr<direct_resampler_conditioner_cb>

Functions

- direct_resampler_conditioner_cb_sptr **direct_resampler_make_conditioner_cb** (double sample_freq_in, double sample_freq_out)

13.287.1 Detailed Description

Nearest neighborhood resampler with std::complex<signed char> input and std::complex<signed char> output.

Author

Luis Esteve, 2011. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file direct_resampler_conditioner_cb.h.

13.288 direct_resampler_conditioner_cb.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file direct_resampler_conditioner_cb.h
00003  * \brief Nearest neighborhood resampler with
```

```
00004  *        std::complex<signed char> input and std::complex<signed char> output
00005  * \author Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CB_H
00020 #define GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CB_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/block.h>
00024 #include <cstdint>
00025
00026 /** \addtogroup Resampler
00027  * \{ */
00028 /** \addtogroup Resampler_gnuradio_blocks
00029  * GNU Radio blocks for input signal resampling
00030  * \{ */
00031
00032
00033 class direct_resampler_conditioner_cb;
00034
00035 using direct_resampler_conditioner_cb_sptr = gnss_shared_ptr<direct_resampler_conditioner_cb>;
00036
00037 direct_resampler_conditioner_cb_sptr direct_resampler_make_conditioner_cb(
00038     double sample_freq_in,
00039     double sample_freq_out);
00040
00041 /*!
00042  * \brief This class implements a direct resampler conditioner for std::complex<signed char>
00043  *
00044  * Direct resampling without interpolation
00045  */
00046 class direct_resampler_conditioner_cb : public gr::block
00047 {
00048 public:
00049     ~direct_resampler_conditioner_cb() = default;
00050
00051     inline unsigned int sample_freq_in() const
00052     {
00053         return d_sample_freq_in;
00054     }
00055
00056     inline unsigned int sample_freq_out() const
00057     {
00058         return d_sample_freq_out;
00059     }
00060
00061     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00062
00063     int general_work(int noutput_items, gr_vector_int &ninput_items,
00064         gr_vector_const_void_star &input_items,
00065         gr_vector_void_star &output_items);
00066
00067 private:
00068     friend direct_resampler_conditioner_cb_sptr direct_resampler_make_conditioner_cb(
00069         double sample_freq_in,
00070         double sample_freq_out);
00071
00072     direct_resampler_conditioner_cb(
00073         double sample_freq_in,
00074         double sample_freq_out);
00075
00076     double d_sample_freq_in;
00077     double d_sample_freq_out;
00078     uint32_t d_phase;
00079     uint32_t d_lphase;
00080     uint32_t d_phase_step;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CB_H
```

13.289 direct_resampler_conditioner_cc.h File Reference

Nearest neighborhood resampler with gr_complex input and gr_complex output.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <cstdint>
```

Classes

- class direct_resampler_conditioner_cc

  This class implements a direct resampler conditioner for complex data.

Typedefs

- using direct_resampler_conditioner_cc_sptr = gnss_shared_ptr<direct_resampler_conditioner_cc>

Functions

- direct_resampler_conditioner_cc_sptr **direct_resampler_make_conditioner_cc** (double sample_freq_in, double sample_freq_out)

13.289.1 Detailed Description

Nearest neighborhood resampler with gr_complex input and gr_complex output.

Author

Luis Esteve, 2011. luis(at)epsilon-formacion.com

This block takes in a signal stream and performs direct resampling. The theory behind this block can be found in Chapter 7.5 of the following book: R. Lyons, Understanding Digital Signal Processing, 3rd ed., Pearson Education, 2010.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file direct_resampler_conditioner_cc.h.

13.290 direct_resampler_conditioner_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file direct_resampler_conditioner_cc.h
00003  *
00004  * \brief Nearest neighborhood resampler with
00005  *        gr_complex input and gr_complex output
00006  * \author Luis Esteve, 2011. luis(at)epsilon-formacion.com
00007  *
00008  * This block takes in a signal stream and performs direct resampling.
00009  * The theory behind this block can be found in Chapter 7.5 of the following
00010  * book:
00011  * R. Lyons, Understanding Digital Signal Processing, 3rd ed., Pearson Education,
00012  * 2010.
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00017  * This file is part of GNSS-SDR.
00018  *
00019  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00020  * SPDX-License-Identifier: GPL-3.0-or-later
00021  *
00022  * -----------------------------------------------------------------------------
00023  */
00024
00025 #ifndef GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CC_H
00026 #define GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CC_H
00027
00028 #include "gnss_block_interface.h"
00029 #include <gnuradio/block.h>
```

```
00030 #include <cstdint>
00031
00032 /** \addtogroup Resampler
00033  * \{ */
00034 /** \addtogroup Resampler_gnuradio_blocks resampler_gr_blocks
00035  * \{ */
00036
00037
00038 class direct_resampler_conditioner_cc;
00039
00040 using direct_resampler_conditioner_cc_sptr = gnss_shared_ptr<direct_resampler_conditioner_cc>;
00041
00042 direct_resampler_conditioner_cc_sptr direct_resampler_make_conditioner_cc(
00043     double sample_freq_in,
00044     double sample_freq_out);
00045
00046 /*!
00047  * \brief This class implements a direct resampler conditioner for complex data
00048  *
00049  * Direct resampling without interpolation
00050  */
00051 class direct_resampler_conditioner_cc : public gr::block
00052 {
00053 public:
00054     ~direct_resampler_conditioner_cc() = default;
00055     inline unsigned int sample_freq_in() const
00056     {
00057         return d_sample_freq_in;
00058     }
00059
00060     inline unsigned int sample_freq_out() const
00061     {
00062         return d_sample_freq_out;
00063     }
00064
00065     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00066
00067     int general_work(int noutput_items, gr_vector_int &ninput_items,
00068         gr_vector_const_void_star &input_items,
00069         gr_vector_void_star &output_items);
00070
00071 private:
00072     friend direct_resampler_conditioner_cc_sptr direct_resampler_make_conditioner_cc(
00073         double sample_freq_in,
00074         double sample_freq_out);
00075
00076     direct_resampler_conditioner_cc(
00077         double sample_freq_in,
00078         double sample_freq_out);
00079
00080     double d_sample_freq_in;   // Sampling frequency of the input signal
00081     double d_sample_freq_out;  // Sampling frequency of the output signal
00082     uint32_t d_phase;
00083     uint32_t d_lphase;
00084     uint32_t d_phase_step;
00085 };
00086
00087
00088 /** \} */
00089 /** \} */
00090 #endif  // GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CC_H
```

13.291 direct_resampler_conditioner_cs.h File Reference

Nearest neighborhood resampler with std::complex<short> input and std::complex<short> output.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <cstdint>
```

Classes

- class direct_resampler_conditioner_cs

  This class implements a direct resampler conditioner for std::complex<short>.

Typedefs

- using direct_resampler_conditioner_cs_sptr = gnss_shared_ptr<direct_resampler_conditioner_cs>

Functions

- direct_resampler_conditioner_cs_sptr **direct_resampler_make_conditioner_cs** (double sample_freq_in, double sample_freq_out)

13.291.1 Detailed Description

Nearest neighborhood resampler with std::complex<short> input and std::complex<short> output.

Author

Luis Esteve, 2011. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file direct_resampler_conditioner_cs.h.

13.292 direct_resampler_conditioner_cs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file direct_resampler_conditioner_cs.h
00003  * \brief Nearest neighborhood resampler with
00004  *        std::complex<short> input and std::complex<short> output
00005  * \author Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CS_H
00020 #define GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CS_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/block.h>
00024 #include <cstdint>
00025
00026 /** \addtogroup Resampler
00027  * \{ */
00028 /** \addtogroup Resampler_gnuradio_blocks
00029  * \{ */
00030
00031
00032 class direct_resampler_conditioner_cs;
00033
00034 using direct_resampler_conditioner_cs_sptr = gnss_shared_ptr<direct_resampler_conditioner_cs>;
00035
00036 direct_resampler_conditioner_cs_sptr direct_resampler_make_conditioner_cs(
00037     double sample_freq_in,
00038     double sample_freq_out);
00039
00040 /*!
00041  * \brief This class implements a direct resampler conditioner for std::complex<short>
00042  *
00043  * Direct resampling without interpolation
00044  */
00045 class direct_resampler_conditioner_cs : public gr::block
00046 {
00047 public:
00048     ~direct_resampler_conditioner_cs() = default;
00049
00050     inline unsigned int sample_freq_in() const
00051     {
00052         return d_sample_freq_in;
00053     }
00054
00055     inline unsigned int sample_freq_out() const
00056     {
00057         return d_sample_freq_out;
00058     }
00059
```

```
00060     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00061
00062     int general_work(int noutput_items, gr_vector_int &ninput_items,
00063         gr_vector_const_void_star &input_items,
00064         gr_vector_void_star &output_items);
00065
00066 private:
00067     friend direct_resampler_conditioner_cs_sptr direct_resampler_make_conditioner_cs(
00068         double sample_freq_in,
00069         double sample_freq_out);
00070
00071     direct_resampler_conditioner_cs(
00072         double sample_freq_in,
00073         double sample_freq_out);
00074
00075     double d_sample_freq_in;
00076     double d_sample_freq_out;
00077     uint32_t d_phase;
00078     uint32_t d_lphase;
00079     uint32_t d_phase_step;
00080 };
00081
00082
00083 /** \} */
00084 /** \} */
00085 #endif  // GNSS_SDR_DIRECT_RESAMPLER_CONDITIONER_CS_H
```

13.293 signal_generator.h File Reference

Adapter of a class that generates synthesized GNSS signal.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "signal_generator_c.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/vector_to_stream.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class SignalGenerator

  This class generates synthesized GNSS signal.

13.293.1 Detailed Description

Adapter of a class that generates synthesized GNSS signal.

Author

Marc Molina, 2013. marc.molina.pena@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file signal_generator.h.

13.294 signal_generator.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_generator.h
00003  * \brief Adapter of a class that generates synthesized GNSS signal.
00004  * \author Marc Molina, 2013. marc.molina.pena@gmail.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
```

```
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_SIGNAL_GENERATOR_H
00020 #define GNSS_SDR_SIGNAL_GENERATOR_H
00021
00022 #include "concurrent_queue.h"
00023 #include "gnss_block_interface.h"
00024 #include "signal_generator_c.h"
00025 #include <gnuradio/blocks/file_sink.h>
00026 #include <gnuradio/blocks/vector_to_stream.h>
00027 #include <gnuradio/hier_block2.h>
00028 #include <pmt/pmt.h>
00029 #include <memory>
00030 #include <string>
00031 #include <vector>
00032
00033 class ConfigurationInterface;
00034
00035 /*!
00036  * \brief This class generates synthesized GNSS signal.
00037  *
00038  */
00039 class SignalGenerator : public GNSSBlockInterface
00040 {
00041 public:
00042     SignalGenerator(const ConfigurationInterface* configuration,
00043         const std::string& role, unsigned int in_stream,
00044         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00045
00046     ~SignalGenerator() = default;
00047
00048     inline std::string role() override
00049     {
00050         return role_;
00051     }
00052
00053     /*!
00054      * \brief Returns "GNSSSignalGenerator".
00055      */
00056     inline std::string implementation() override
00057     {
00058         return "GNSSSignalGenerator";
00059     }
00060
00061     inline size_t item_size() override
00062     {
00063         return item_size_;
00064     }
00065
00066     void connect(gr::top_block_sptr top_block) override;
00067     void disconnect(gr::top_block_sptr top_block) override;
00068     gr::basic_block_sptr get_left_block() override;
00069     gr::basic_block_sptr get_right_block() override;
00070
00071 private:
00072     gnss_shared_ptr<gr::block> gen_source_;
00073     gr::blocks::vector_to_stream::sptr vector_to_stream_;
00074     gr::blocks::file_sink::sptr file_sink_;
00075     std::string role_;
00076     std::string item_type_;
00077     std::string dump_filename_;
00078     size_t item_size_;
00079     unsigned int in_stream_;
00080     unsigned int out_stream_;
00081     bool dump_;
00082 };
00083
00084 #endif  // GNSS_SDR_SIGNAL_GENERATOR_H
```

13.295 signal_generator_c.h File Reference

GNU Radio source block that generates synthesized GNSS signal.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
```

```
#include <random>
#include <string>
#include <vector>
```

Classes

- class signal_generator_c

  This class generates synthesized GNSS signal.

Typedefs

- using signal_generator_c_sptr = gnss_shared_ptr<signal_generator_c>

Functions

- signal_generator_c_sptr signal_make_generator_c (const std::vector< std::string > &signal1, const std::vector< std::string > &system, const std::vector< unsigned int > &PRN, const std::vector< float > &CN0_dB, const std::vector< float > &doppler_Hz, const std::vector< unsigned int > &delay_chips, const std::vector< unsigned int > &delay_sec, bool data_flag, bool noise_flag, unsigned int fs_in, unsigned int vector_length, float BW_BB)

  Return a shared_ptr to a new instance of gen_source.

13.295.1 Detailed Description

GNU Radio source block that generates synthesized GNSS signal.

Author

Marc Molina, 2013. marc.molina.pena@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file signal_generator_c.h.

13.295.2 Typedef Documentation

13.295.2.1 signal_generator_c_sptr

```
using signal_generator_c_sptr = gnss_shared_ptr<signal_generator_c>
```

Definition at line 29 of file signal_generator_c.h.

13.295.3 Function Documentation

13.295.3.1 signal_make_generator_c()

```
signal_generator_c_sptr signal_make_generator_c (
            const std::vector< std::string > & signal1,
            const std::vector< std::string > & system,
            const std::vector< unsigned int > & PRN,
            const std::vector< float > & CN0_dB,
            const std::vector< float > & doppler_Hz,
            const std::vector< unsigned int > & delay_chips,
            const std::vector< unsigned int > & delay_sec,
            bool data_flag,
            bool noise_flag,
            unsigned int fs_in,
            unsigned int vector_length,
            float BW_BB)
```

Return a shared_ptr to a new instance of gen_source.

To avoid accidental use of raw pointers, gen_source's constructor is private. signal_make_generator_c is the public interface for creating new instances.
References signal_make_generator_c().
Referenced by signal_make_generator_c().

13.296 signal_generator_c.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_generator_c.h
00003  * \brief GNU Radio source block that generates synthesized GNSS signal.
00004  * \author Marc Molina, 2013. marc.molina.pena@gmail.com
00005  *
00006  * -------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SIGNAL_GENERATOR_C_H
00018 #define GNSS_SDR_SIGNAL_GENERATOR_C_H
00019
00020 #include "gnss_block_interface.h"
00021 #include <gnuradio/block.h>
00022 #include <random>
00023 #include <string>
00024 #include <vector>
00025
00026
00027 class signal_generator_c;
00028
00029 using signal_generator_c_sptr = gnss_shared_ptr<signal_generator_c>;
00030
00031 /*!
00032  * \brief Return a shared_ptr to a new instance of gen_source.
00033  *
00034  * To avoid accidental use of raw pointers, gen_source's
00035  * constructor is private. signal_make_generator_c is the public
00036  * interface for creating new instances.
00037  */
00038 signal_generator_c_sptr signal_make_generator_c(
00039     const std::vector<std::string> &signal1,
00040     const std::vector<std::string> &system,
00041     const std::vector<unsigned int> &PRN,
00042     const std::vector<float> &CN0_dB,
00043     const std::vector<float> &doppler_Hz,
00044     const std::vector<unsigned int> &delay_chips,
00045     const std::vector<unsigned int> &delay_sec,
00046     bool data_flag,
00047     bool noise_flag,
00048     unsigned int fs_in,
00049     unsigned int vector_length,
00050     float BW_BB);
00051
00052 /*!
00053  * \brief This class generates synthesized GNSS signal.
00054  * \ingroup block
00055  *
00056  * \sa gen_source for a version that subclasses gr_block.
00057  */
00058 class signal_generator_c : public gr::block
00059 {
00060 public:
00061     ~signal_generator_c() = default;  // public destructor
00062
00063     // Where all the action really happens
00064     int general_work(int noutput_items,
00065         gr_vector_int &ninput_items,
00066         gr_vector_const_void_star &input_items,
00067         gr_vector_void_star &output_items);
00068
00069 private:
00070     friend signal_generator_c_sptr signal_make_generator_c(
00071         const std::vector<std::string> &signal1,
00072         const std::vector<std::string> &system,
00073         const std::vector<unsigned int> &PRN,
00074         const std::vector<float> &CN0_dB,
```

```
00075         const std::vector<float> &doppler_Hz,
00076         const std::vector<unsigned int> &delay_chips,
00077         const std::vector<unsigned int> &delay_sec,
00078         bool data_flag,
00079         bool noise_flag,
00080         unsigned int fs_in,
00081         unsigned int vector_length,
00082         float BW_BB);
00083
00084     signal_generator_c(
00085         std::vector<std::string> signal1,
00086         std::vector<std::string> system,
00087         const std::vector<unsigned int> &PRN,
00088         std::vector<float> CN0_dB,
00089         std::vector<float> doppler_Hz,
00090         std::vector<unsigned int> delay_chips,
00091         std::vector<unsigned int> delay_sec,
00092         bool data_flag,
00093         bool noise_flag,
00094         unsigned int fs_in,
00095         unsigned int vector_length,
00096         float BW_BB);
00097
00098     void init();
00099
00100     void generate_codes();
00101
00102     std::random_device r;
00103     std::uniform_int_distribution<int> uniform_dist;
00104     std::normal_distribution<float> normal_dist;
00105     std::vector<std::string> signal_;
00106     std::vector<std::string> system_;
00107     std::vector<std::vector<gr_complex» sampled_code_data_;
00108     std::vector<std::vector<gr_complex» sampled_code_pilot_;
00109     std::vector<gr_complex> current_data_bits_;
00110     std::vector<gr_complex> complex_phase_;
00111     std::vector<float> CN0_dB_;
00112     std::vector<float> doppler_Hz_;
00113     std::vector<float> start_phase_rad_;
00114     std::vector<unsigned int> PRN_;
00115     std::vector<unsigned int> delay_chips_;
00116     std::vector<unsigned int> delay_sec_;
00117     std::vector<unsigned int> samples_per_code_;
00118     std::vector<unsigned int> num_of_codes_per_vector_;
00119     std::vector<unsigned int> data_bit_duration_ms_;
00120     std::vector<unsigned int> ms_counter_;
00121     std::vector<signed int> current_data_bit_int_;
00122     std::vector<signed int> data_modulation_;
00123     std::vector<signed int> pilot_modulation_;
00124     float BW_BB_;
00125     unsigned int work_counter_{};
00126     unsigned int fs_in_;
00127     unsigned int num_sats_;
00128     unsigned int vector_length_;
00129     bool data_flag_;
00130     bool noise_flag_;
00131 };
00132
00133 #endif  // GNSS_SDR_SIGNAL_GENERATOR_C_H
```

13.297 ad936x_custom_signal_source.h File Reference

A direct IIO custom front-end gnss-sdr signal source for the AD936x AD front-end family with special FPGA custom functionalities.

```
#include "ad936x_iio_source.h"
#include "concurrent_queue.h"
#include "conjugate_cc.h"
#include "signal_source_base.h"
#include "unpack_byte_2bit_cpx_samples.h"
#include "unpack_byte_4bit_samples.h"
#include "unpack_short_byte_samples.h"
#include <gnuradio/blocks/char_to_short.h>
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/delay.h>
#include <gnuradio/blocks/interleaved_short_to_complex.h>
#include <pmt/pmt.h>
#include <cstdint>
```

```
#include <memory>
#include <stdexcept>
#include <string>
#include <vector>
```

Classes

- class Ad936xCustomSignalSource

  This class instantiates the Ad936xCustom gnuradio signal source. It has support also for a customized Ad936x↩ Custom firmware and signal source to support PPS samplestamp reading.

13.297.1 Detailed Description

A direct IIO custom front-end gnss-sdr signal source for the AD936x AD front-end family with special FPGA custom functionalities.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ad936x_custom_signal_source.h.

13.298 ad936x_custom_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ad936x_custom_signal_source.h
00003  * \brief A direct IIO custom front-end gnss-sdr signal source for the AD936x AD front-end family with
      special FPGA custom functionalities.
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_Ad936xCustom_SIGNAL_SOURCE_H
00018 #define GNSS_SDR_Ad936xCustom_SIGNAL_SOURCE_H
00019
00020 #include "ad936x_iio_source.h"
00021 #include "concurrent_queue.h"
00022 #include "conjugate_cc.h"
00023 #include "signal_source_base.h"
00024 #include "unpack_byte_2bit_cpx_samples.h"
00025 #include "unpack_byte_4bit_samples.h"
00026 #include "unpack_short_byte_samples.h"
00027 #include <gnuradio/blocks/char_to_short.h>
00028 #include <gnuradio/blocks/file_sink.h>
00029 // #include <gnuradio/blocks/interleaved_char_to_complex.h>
00030 #include <gnuradio/blocks/delay.h>
00031 #include <gnuradio/blocks/interleaved_short_to_complex.h>
00032 #include <pmt/pmt.h>
00033 #include <cstdint>
00034 #include <memory>
00035 #include <stdexcept>
00036 #include <string>
00037 #include <vector>
00038
00039 /** \addtogroup Signal_Source
00040  * \{ */
00041 /** \addtogroup Signal_Source_adapters
00042  * \{ */
00043
00044
00045 class ConfigurationInterface;
00046
```

```
00047 /*!
00048  * \brief This class instantiates the Ad936xCustom gnuradio signal source.
00049  * It has support also for a customized Ad936xCustom firmware and signal source to support PPS
      samplestamp reading.
00050  */
00051 class Ad936xCustomSignalSource : public SignalSourceBase
00052 {
00053 public:
00054     Ad936xCustomSignalSource(const ConfigurationInterface* configuration,
00055         const std::string& role, unsigned int in_stream,
00056         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00057
00058     ~Ad936xCustomSignalSource() = default;
00059
00060     inline size_t item_size() override
00061     {
00062         return item_size_;
00063     }
00064
00065     void connect(gr::top_block_sptr top_block) override;
00066     void disconnect(gr::top_block_sptr top_block) override;
00067     gr::basic_block_sptr get_left_block() override;
00068     gr::basic_block_sptr get_right_block() override;
00069     gr::basic_block_sptr get_right_block(int RF_channel) override;
00070
00071 private:
00072     unsigned int in_stream_;
00073     unsigned int out_stream_;
00074     gr::block_sptr ad936x_iio_source;
00075     std::vector<gr::blocks::file_sink::sptr> sink_;
00076     std::vector<std::string> filename_vec_;
00077
00078     gr::blocks::delay::sptr gr_delay;
00079     std::vector<gr::blocks::char_to_short::sptr> gr_char_to_short_;
00080     std::vector<gr::blocks::interleaved_short_to_complex::sptr> gr_interleaved_short_to_complex_;
00081     //    std::vector<gr::blocks::interleaved_char_to_complex::sptr> gr_interleaved_char_to_complex_;
00082     std::vector<unpack_short_byte_samples_sptr> unpack_short_byte;
00083     std::vector<unpack_byte_4bit_samples_sptr> unpack_byte_fourbits;
00084     std::vector<unpack_byte_2bit_cpx_samples_sptr> unpack_byte_twobits;
00085
00086     std::string item_type_;
00087     size_t item_size_;
00088     int64_t samples_;
00089     bool dump_;
00090     std::string dump_filename_;
00091
00092     // Front-end settings
00093     std::string pluto_uri_;
00094     std::string board_type_;
00095     long long sample_rate_;
00096     long long bandwidth_;
00097     long long freq_;
00098     long long freq_2ch;
00099     std::string rf_port_select_;
00100     std::string rf_filter;
00101     std::string gain_mode_rx0_;
00102     std::string gain_mode_rx1_;
00103     double rf_gain_rx0_;
00104     double rf_gain_rx1_;
00105     bool enable_ch0;
00106     bool enable_ch1;
00107     bool PPS_mode_;
00108     std::string fe_ip_;
00109     int fe_ctlport_;
00110     int ssize_;
00111     int bshift_;
00112     bool spattern_;
00113     bool inverted_spectrum_ch0_;
00114     bool inverted_spectrum_ch1_;
00115     double lo_attenuation_db_;
00116     bool high_side_lo_;
00117     int tx_lo_channel_;
00118     double rx0_to_rx1_delay_ns_;
00119     bool delay_enabled;
00120     bool apply_delay_on_rx0;
00121
00122     std::vector<bool> inverted_spectrum_vec;
00123     int n_channels;
00124 };
00125
00126
00127 /** \} */
00128 /** \} */
00129 #endif  // GNSS_SDR_Ad936xCustom_SIGNAL_SOURCE_H
```

13.299 adrv9361_z7035_signal_source_fpga.h File Reference

Signal source for the Analog Devices ADRV9361-Z7035 evaluation board directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file.

```
#include "concurrent_queue.h"
#include "fpga_buffer_monitor.h"
#include "fpga_dynamic_bit_selection.h"
#include "fpga_switch.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
#include <thread>
```

Classes

- class Adrv9361z7035SignalSourceFPGA

13.299.1 Detailed Description

Signal source for the Analog Devices ADRV9361-Z7035 evaluation board directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file adrv9361_z7035_signal_source_fpga.h.

13.300 adrv9361_z7035_signal_source_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file adrv9361_z7035_signal_source_fpga.h
00003  * \brief Signal source for the Analog Devices ADRV9361-Z7035 evaluation board
00004  * directly connected to the FPGA accelerators.
00005  * This source implements only the AD9361 control. It is NOT compatible with
00006  * conventional SDR acquisition and tracking blocks.
00007  * Please use the fmcomms2 source if conventional SDR acquisition and tracking
00008  * is selected in the configuration file.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_ADRV9361_Z7035_SIGNAL_SOURCE_FPGA_H
00022 #define GNSS_SDR_ADRV9361_Z7035_SIGNAL_SOURCE_FPGA_H
00023
00024 #include "concurrent_queue.h"
00025 #include "fpga_buffer_monitor.h"
00026 #include "fpga_dynamic_bit_selection.h"
00027 #include "fpga_switch.h"
00028 #include "gnss_block_interface.h"
00029 #include "signal_source_base.h"
00030 #include <pmt/pmt.h>
00031 #include <cstdint>
00032 #include <memory>
```

```
00033 #include <mutex>
00034 #include <string>
00035 #include <thread>
00036
00037
00038 /** \addtogroup Signal_Source
00039  * \{ */
00040 /** \addtogroup Signal_Source_adapters
00041  * \{ */
00042
00043
00044 class ConfigurationInterface;
00045
00046 class Adrv9361z7035SignalSourceFPGA : public SignalSourceBase
00047 {
00048 public:
00049     Adrv9361z7035SignalSourceFPGA(const ConfigurationInterface *configuration,
00050         const std::string &role, unsigned int in_stream,
00051         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t> *queue);
00052
00053     ~Adrv9361z7035SignalSourceFPGA();
00054
00055     inline size_t item_size() override
00056     {
00057         return item_size_;
00058     }
00059
00060     void connect(gr::top_block_sptr top_block) override;
00061     void disconnect(gr::top_block_sptr top_block) override;
00062     gr::basic_block_sptr get_left_block() override;
00063     gr::basic_block_sptr get_right_block() override;
00064
00065 private:
00066     const std::string DEFAULT_DUMP_FILENAME = std::string("FPGA_buffer_monitor_dump.dat");
00067     const std::string DEFAULT_RF_PORT_SELECT = std::string("A_BALANCED");
00068     const std::string DEFAULT_GAIN_MODE = std::string("slow_attack");
00069     const double DEFAULT_TX_ATTENUATION_dB = -10.0;
00070     const double DEFAULT_MANUAL_GAIN_RX1 = 64.0;
00071     const double DEFAULT_MANUAL_GAIN_RX2 = 64.0;
00072     const uint64_t DEFAULT_BANDWIDTH = 12500000;
00073
00074     // perform dynamic bit selection every 500 ms by default
00075     const uint32_t GAIN_CONTROL_PERIOD_ms = 500;
00076     // check buffer overflow and perform buffer monitoring every 1s by default
00077     const uint32_t BUFFER_MONITOR_PERIOD_ms = 1000;
00078     // buffer overflow and buffer monitoring initial delay
00079     const uint32_t BUFFER_MONITOR_INITIAL_DELAY_ms = 2000;
00080     const int32_t REAL_TIME_MODE = 2;
00081
00082     void run_dynamic_bit_selection_process();
00083     void run_buffer_monitor_process();
00084
00085     mutable std::mutex dynamic_bit_selection_mutex;
00086     mutable std::mutex buffer_monitor_mutex;
00087
00088     std::thread thread_dynamic_bit_selection;
00089     std::thread thread_buffer_monitor;
00090
00091     std::shared_ptr<Fpga_Switch> switch_fpga;
00092     std::shared_ptr<Fpga_dynamic_bit_selection> dynamic_bit_selection_fpga;
00093     std::shared_ptr<Fpga_buffer_monitor> buffer_monitor_fpga;
00094
00095     std::string gain_mode_rx1_;
00096     std::string gain_mode_rx2_;
00097     std::string rf_port_select_;
00098     std::string filter_file_;
00099     std::string filter_source_;
00100     std::string filter_filename_;
00101
00102     double rf_gain_rx1_;
00103     double rf_gain_rx2_;
00104     double scale_dds_dbfs_;
00105     double phase_dds_deg_;
00106     double tx_attenuation_db_;
00107
00108     uint64_t freq0_;  // frequency of local oscillator for ADRV9361-A 0
00109     uint64_t sample_rate_;
00110     uint64_t bandwidth_;
00111     uint64_t freq_dds_tx_hz_;
00112     uint64_t freq_rf_tx_hz_;
00113     uint64_t tx_bandwidth_;
00114
00115     float Fpass_;
00116     float Fstop_;
00117     uint32_t in_stream_;
00118     uint32_t out_stream_;
00119
```

```
00120     size_t item_size_;
00121
00122     bool enable_dds_lo_;
00123     bool quadrature_;
00124     bool rf_dc_;
00125     bool bb_dc_;
00126     bool rx1_enable_;
00127     bool rx2_enable_;
00128     bool enable_dynamic_bit_selection_;
00129     bool enable_ovf_check_buffer_monitor_active_;
00130     bool dump_;
00131     bool rf_shutdown_;
00132 };
00133
00134
00135 /** \} */
00136 /** \} */
00137 #endif  // GNSS_SDR_ADRV9361_Z7035_SIGNAL_SOURCE_FPGA_H
```

13.301 bladerf_signal_source.h File Reference

Signal source for the Nuand bladeRF family of front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), using libbladeRF directly.

```
#include "bladerf_source.h"
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
```

Classes

- class BladerfSignalSource

  This class instantiates the bladeRF signal source, streaming RX samples directly through libbladeRF (no gr-osmosdr or other intermediate driver required).

13.301.1 Detailed Description

Signal source for the Nuand bladeRF family of front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), using libbladeRF directly.

Author

Oleksandr Suvorov, 2026.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file bladerf_signal_source.h.

13.302 bladerf_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file bladerf_signal_source.h
00003  * \brief Signal source for the Nuand bladeRF family of front-ends (bladeRF
00004  * x40, x115, and bladeRF 2.0 Micro xA4/xA9), using libbladeRF directly.
00005  * \author Oleksandr Suvorov, 2026.
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
```

```
00016  */
00017
00018 #ifndef GNSS_SDR_BLADERF_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_BLADERF_SIGNAL_SOURCE_H
00020
00021 #include "bladerf_source.h"
00022 #include "concurrent_queue.h"
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/file_sink.h>
00025 #include <pmt/pmt.h>
00026 #include <cstdint>
00027 #include <string>
00028
00029 /** \addtogroup Signal_Source
00030  * \{ */
00031 /** \addtogroup Signal_Source_adapters
00032  * \{ */
00033
00034
00035 class ConfigurationInterface;
00036
00037 /*!
00038  * \brief This class instantiates the bladeRF signal source, streaming RX
00039  * samples directly through libbladeRF (no gr-osmosdr or other intermediate
00040  * driver required).
00041  */
00042 class BladerfSignalSource : public SignalSourceBase
00043 {
00044 public:
00045     BladerfSignalSource(const ConfigurationInterface* configuration,
00046         const std::string& role, unsigned int in_stream,
00047         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00048
00049     ~BladerfSignalSource() = default;
00050
00051     inline size_t item_size() override
00052     {
00053         return item_size_;
00054     }
00055
00056     void connect(gr::top_block_sptr top_block) override;
00057     void disconnect(gr::top_block_sptr top_block) override;
00058     gr::basic_block_sptr get_left_block() override;
00059     gr::basic_block_sptr get_right_block() override;
00060
00061 private:
00062     bladerf_source_sptr bladerf_source_;
00063     gnss_shared_ptr<gr::block> valve_;
00064     gr::blocks::file_sink::sptr file_sink_;
00065
00066     std::string item_type_;
00067     std::string dump_filename_;
00068     std::string bladerf_args_;
00069     std::string device_serial_;
00070
00071     double sample_rate_;
00072     double freq_;
00073     double bandwidth_;
00074     double gain_;
00075     size_t item_size_;
00076     int64_t samples_;
00077
00078     unsigned int in_stream_;
00079     unsigned int out_stream_;
00080     unsigned int num_buffers_;
00081     unsigned int buffer_size_;
00082     unsigned int num_transfers_;
00083     unsigned int stream_timeout_ms_;
00084
00085     bool agc_enabled_;
00086     bool rx_bias_tee_;
00087     bool dump_;
00088 };
00089
00090
00091 /** \} */
00092 /** \} */
00093 #endif  // GNSS_SDR_BLADERF_SIGNAL_SOURCE_H
```

13.303 custom_udp_signal_source.h File Reference

Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap).

```
#include "concurrent_queue.h"
#include "gr_complex_ip_packet_source.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/null_sink.h>
#include <pmt/pmt.h>
#include <stdexcept>
#include <string>
#include <vector>
```

Classes

- class CustomUDPSignalSource

  This class reads from UDP packets, which streams interleaved I/Q samples over a network.

13.303.1 Detailed Description

Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap).

13.303.1.1 autotoc_md299

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file custom_udp_signal_source.h.

13.304 custom_udp_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file custom_udp_signal_source.h
00003  * \brief Receives ip frames containing samples in UDP frame encapsulation
00004  * using a high performance packet capture library (libpcap)
00005  * \author Javier Arribas jarribas (at) cttc.es
00006  * -------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_CUSTOM_UDP_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_CUSTOM_UDP_SIGNAL_SOURCE_H
00020
00021 #include "concurrent_queue.h"
00022 #include "gr_complex_ip_packet_source.h"
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/file_sink.h>
00025 #include <gnuradio/blocks/null_sink.h>
00026 #include <pmt/pmt.h>
00027 #include <stdexcept>
00028 #include <string>
00029 #include <vector>
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
```

```
00038
00039 /*!
00040  * \brief This class reads from UDP packets, which streams interleaved
00041  * I/Q samples over a network.
00042  */
00043 class CustomUDPSignalSource : public SignalSourceBase
00044 {
00045 public:
00046     CustomUDPSignalSource(const ConfigurationInterface* configuration,
00047         const std::string& role, unsigned int in_stream,
00048         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00049
00050     ~CustomUDPSignalSource() = default;
00051
00052     inline size_t item_size() override
00053     {
00054         return item_size_;
00055     }
00056
00057     void connect(gr::top_block_sptr top_block) override;
00058     void disconnect(gr::top_block_sptr top_block) override;
00059     gr::basic_block_sptr get_left_block() override;
00060     gr::basic_block_sptr get_right_block() override;
00061     gr::basic_block_sptr get_right_block(int RF_channel) override;
00062
00063 private:
00064     Gr_Complex_Ip_Packet_Source::sptr udp_gnss_rx_source_;
00065     std::vector<gnss_shared_ptr<gr::block» null_sinks_;
00066     std::vector<gnss_shared_ptr<gr::block» file_sink_;
00067
00068     std::string item_type_;
00069     std::string dump_filename_;
00070
00071     size_t item_size_;
00072
00073     int RF_channels_;
00074     int channels_in_udp_;
00075     unsigned int in_stream_;
00076     unsigned int out_stream_;
00077     bool IQ_swap_;
00078     bool dump_;
00079 };
00080
00081
00082 /** \} */
00083 /** \} */
00084 #endif  // GNSS_SDR_CUSTOM_UDP_SIGNAL_SOURCE_H
```

13.305 dma_signal_source_fpga.h File Reference

signal source for a DMA connected directly to FPGA accelerators. This source implements only the DMA control. It is NOT compatible with conventional SDR acquisition and tracking blocks.

```
#include "concurrent_queue.h"
#include "fpga_dma-proxy.h"
#include "fpga_dynamic_bit_selection.h"
#include "fpga_switch.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
#include <thread>
```

Classes

- class DMASignalSourceFPGA

13.305.1 Detailed Description

signal source for a DMA connected directly to FPGA accelerators. This source implements only the DMA control. It is NOT compatible with conventional SDR acquisition and tracking blocks.

Author

Marc Majoral, mmajoral(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file dma_signal_source_fpga.h.

13.306 dma_signal_source_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dma_signal_source_fpga.h
00003  * \brief signal source for a DMA connected directly to FPGA accelerators.
00004  * This source implements only the DMA control. It is NOT compatible with
00005  * conventional SDR acquisition and tracking blocks.
00006  * \author Marc Majoral, mmajoral(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DMA_SIGNAL_SOURCE_FPGA_H
00020 #define GNSS_SDR_DMA_SIGNAL_SOURCE_FPGA_H
00021
00022 #include "concurrent_queue.h"
00023 #include "fpga_dma-proxy.h"
00024 #include "fpga_dynamic_bit_selection.h"
00025 #include "fpga_switch.h"
00026 #include "gnss_block_interface.h"
00027 #include "signal_source_base.h"
00028 #include <pmt/pmt.h>
00029 #include <cstdint>
00030 #include <memory>
00031 #include <mutex>
00032 #include <string>
00033 #include <thread>
00034
00035
00036 /** \addtogroup Signal_Source
00037  * \{ */
00038 /** \addtogroup Signal_Source_adapters
00039  * \{ */
00040
00041
00042 class ConfigurationInterface;
00043
00044 class DMASignalSourceFPGA : public SignalSourceBase
00045 {
00046 public:
00047     DMASignalSourceFPGA(const ConfigurationInterface *configuration,
00048         const std::string &role, unsigned int in_stream,
00049         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t> *queue);
00050
00051     ~DMASignalSourceFPGA();
00052
00053     void start() override;
00054
00055     inline size_t item_size() override
00056     {
00057         return item_size_;
00058     }
00059
00060     void connect(gr::top_block_sptr top_block) override;
00061     void disconnect(gr::top_block_sptr top_block) override;
00062     gr::basic_block_sptr get_left_block() override;
00063     gr::basic_block_sptr get_right_block() override;
00064
00065 private:
```

```
00066     const std::string EMPTY_STRING;
00067     const uint64_t DEFAULT_BANDWIDTH = 12500000;
00068     const uint32_t GAIN_CONTROL_PERIOD_ms = 500;  // perform dynamic bit selection every 500 ms by
      default
00069     const int32_t POST_PROCESSING_MODE = 0;
00070     const uint32_t IQ_COMPONENTS_PER_SAMPLE = 2;
00071     static constexpr uint32_t IQ_COMPONENTS_PER_DMA_FRAME = 4;  // Two signals, each with I and Q.
00072
00073     uint64_t get_available_items(const std::string &filename) const;
00074
00075     void run_DMA_process(const std::string &filename0,
00076         const std::string &filename1,
00077         uint64_t &bytes_to_skip,
00078         size_t &item_size,
00079         int64_t &samples,
00080         bool &repeat,
00081         uint32_t &dma_buff_offset_pos,
00082         Concurrent_Queue<pmt::pmt_t> *queue);
00083
00084     void run_dynamic_bit_selection_process();
00085
00086     std::thread thread_file_to_dma;
00087     std::thread thread_dynamic_bit_selection;
00088
00089     std::shared_ptr<Fpga_Switch> switch_fpga;
00090     std::shared_ptr<Fpga_dynamic_bit_selection> dynamic_bit_selection_fpga;
00091     std::shared_ptr<Fpga_DMA> dma_fpga;
00092
00093     std::mutex dma_mutex;
00094     std::mutex dynamic_bit_selection_mutex;
00095
00096     Concurrent_Queue<pmt::pmt_t> *queue_;
00097
00098     std::string filename0_;
00099     std::string filename1_;
00100
00101     uint64_t sample_rate_;
00102     uint64_t bytes_to_skip_;
00103     int64_t samples_;
00104     uint32_t num_input_files_;
00105     uint32_t dma_buff_offset_pos_;
00106     uint32_t in_stream_;
00107     uint32_t out_stream_;
00108     size_t item_size_;
00109
00110     bool enable_DMA_;
00111     bool rx1_enable_;
00112     bool rx2_enable_;
00113     bool enable_dynamic_bit_selection_;
00114     bool repeat_;
00115 };
00116
00117
00118 /** \} */
00119 /** \} */
00120 #endif  // GNSS_SDR_DMA_SIGNAL_SOURCE_FPGA_H
```

13.307 evk1029_signal_source.h File Reference

SAPHYRION EVK1029 signal source. Reads a continuous, header-less raw capture file (OBA-encoded 4-bit samples) directly, without going through the generic XML-metadata-driven ION_GSMS_Signal_Source path.

```
#include "concurrent_queue.h"
#include "evk1029_source.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/throttle.h>
#include <cstdint>
#include <string>
```

Classes

- class Evk1029SignalSource

  This class reads a SAPHYRION EVK1029 raw capture file.

13.307.1 Detailed Description

SAPHYRION EVK1029 signal source. Reads a continuous, header-less raw capture file (OBA-encoded 4-bit samples) directly, without going through the generic XML-metadata-driven ION_GSMS_Signal_Source path.
Successor of the old EVK10x9_signal_source: same overall structure (dedicated file reader + optional throttle + optional dump), updated for the current EVK1029 capture format (see evk1029_source.h).
Set role + ".RF_channels" to N > 1 (e.g. for a multi-band setup sharing a single IF capture) to feed N Signal Conditioners from ONE underlying Evk1029Source instance/read, instead of instantiating N separate SignalSourceN blocks each reading the file independently. Evk1029Source has a single output port (see its own header); gnss↩_flowgraph.cc's generic signal-source-to-conditioner wiring detects this (output_signature()->max_streams() == 1) and connects each of the N Signal Conditioners to that same port, so all of them read the same sample stream and GNU Radio keeps them within one output buffer of each other. The optional throttle sits between the source and that shared port, so .enable_throttle_control works for any RF_channels value. Set role + ".seconds_to_skip" to start reading at a given time into the capture (converted to a byte position using the raw sampling frequency).
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file evk1029_signal_source.h.

13.308 evk1029_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file evk1029_signal_source.h
00003  * \brief SAPHYRION EVK1029 signal source. Reads a continuous, header-less
00004  * raw capture file (OBA-encoded 4-bit samples) directly, without going
00005  * through the generic XML-metadata-driven ION_GSMS_Signal_Source path.
00006  *
00007  * Successor of the old EVK10x9_signal_source: same overall structure
00008  * (dedicated file reader + optional throttle + optional dump), updated
00009  * for the current EVK1029 capture format (see evk1029_source.h).
00010  *
00011  * Set role + ".RF_channels" to N > 1 (e.g. for a multi-band setup sharing a
00012  * single IF capture) to feed N Signal Conditioners from ONE underlying
00013  * Evk1029Source instance/read, instead of instantiating N separate
00014  * SignalSourceN blocks each reading the file independently. Evk1029Source
00015  * has a single output port (see its own header); gnss_flowgraph.cc's
00016  * generic signal-source-to-conditioner wiring detects this
00017  * (output_signature()->max_streams() == 1) and connects each of the N
00018  * Signal Conditioners to that same port, so all of them read the same
00019  * sample stream and GNU Radio keeps them within one output buffer of each
00020  * other. The optional throttle sits between the source and that shared
00021  * port, so .enable_throttle_control works for any RF_channels value.
00022  * Set role + ".seconds_to_skip" to start reading at a given time into the
00023  * capture (converted to a byte position using the raw sampling frequency).
00024  *
00025  * -----------------------------------------------------------------------------
00026  *
00027  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00028  * This file is part of GNSS-SDR.
00029  *
00030  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00031  * SPDX-License-Identifier: GPL-3.0-or-later
00032  *
00033  * -----------------------------------------------------------------------------
00034  */
00035
00036 #ifndef GNSS_SDR_EVK1029_SIGNAL_SOURCE_H
00037 #define GNSS_SDR_EVK1029_SIGNAL_SOURCE_H
00038
00039 #include "concurrent_queue.h"
00040 #include "evk1029_source.h"
00041 #include "signal_source_base.h"
00042 #include <gnuradio/blocks/file_sink.h>
00043 #include <gnuradio/blocks/throttle.h>
00044 #include <cstdint>
00045 #include <string>
00046
00047 class ConfigurationInterface;
00048
00049 /** \addtogroup Signal_Source
00050  * \{ */
00051 /** \addtogroup Signal_Source_adapters
00052  * \{ */
00053
00054
00055 /*!
00056  * \brief This class reads a SAPHYRION EVK1029 raw capture file.
```

```
00057  */
00058 class Evk1029SignalSource : public SignalSourceBase
00059 {
00060 public:
00061     Evk1029SignalSource(
00062         const ConfigurationInterface* configuration,
00063         const std::string& role,
00064         unsigned int in_stream,
00065         unsigned int out_stream,
00066         Concurrent_Queue<pmt::pmt_t>* queue);
00067
00068     ~Evk1029SignalSource() = default;
00069
00070     inline size_t item_size() override
00071     {
00072         return item_size_;
00073     }
00074
00075     void connect(gr::top_block_sptr top_block) override;
00076     void disconnect(gr::top_block_sptr top_block) override;
00077     gr::basic_block_sptr get_right_block() override;
00078
00079 private:
00080     Evk1029Source_sptr evk1029_source_;
00081     gr::blocks::throttle::sptr throttle_;
00082     gr::blocks::file_sink::sptr file_sink_;
00083
00084     std::string dump_filename_;
00085     bool dump_;
00086
00087     size_t item_size_;
00088     unsigned int rf_channels_;
00089
00090     bool enable_throttle_control_;
00091 };
00092
00093
00094 /** \} */
00095 /** \} */
00096 #endif  // GNSS_SDR_EVK1029_SIGNAL_SOURCE_H
```

13.309 fifo_signal_source.h File Reference

Header file of the class for retrieving samples through a Unix FIFO.

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <pmt/pmt.h>
#include <cstddef>
#include <string>
```

Classes

- class FifoSignalSource

  Class that reads a sample stream from a Unix FIFO.

13.309.1 Detailed Description

Header file of the class for retrieving samples through a Unix FIFO.

Author

Malte Lenhart, 2021. malte.lenhart(at)mailbox.org

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fifo_signal_source.h.

13.310 fifo_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fifo_signal_source.h
00003  *
00004  * \brief Header file of the class for retrieving samples through a Unix FIFO
00005  * \author Malte Lenhart, 2021. malte.lenhart(at)mailbox.org
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_FIFO_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_FIFO_SIGNAL_SOURCE_H
00021
00022 #include "concurrent_queue.h"
00023 #include "signal_source_base.h"
00024 #include <pmt/pmt.h>
00025 #include <cstddef>
00026 #include <string>
00027
00028 /** \addtogroup Signal_Source
00029  * \{ */
00030 /** \addtogroup Signal_Source_adapters
00031  * \{ */
00032
00033 // forward declaration to avoid include in header
00034 class ConfigurationInterface;
00035
00036 //! \brief Class that reads a sample stream from a Unix FIFO.
00037 //!
00038 //! This class supports the following properties:
00039 //!
00040 //!   .filename - the path to the input file
00041 //!             - may be overridden by the -signal_source or -s command-line arguments
00042 //!
00043 //!   .sample_type - data type read out from the FIFO. default ishort ;
00044 //!                - note: not output format. that is always gr_complex
00045 //!
00046 //!   .dump     - whether to archive input data
00047 //!
00048 //!   .dump_filename - if dumping, path to file for output
00049 //!
00050 class FifoSignalSource : public SignalSourceBase
00051 {
00052 public:
00053     FifoSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00054         unsigned int in_streams, unsigned int out_streams,
00055         Concurrent_Queue<pmt::pmt_t>* queue);
00056
00057     ~FifoSignalSource() = default;
00058
00059     //! override methods from GNSSBlockInterface
00060     void connect(gr::top_block_sptr top_block) override;
00061     void disconnect(gr::top_block_sptr top_block) override;
00062     size_t item_size() override;
00063     gr::basic_block_sptr get_left_block() override;
00064     gr::basic_block_sptr get_right_block() override;
00065
00066 protected:
00067 private:
00068     //! output size - always gr_complex
00069     const size_t item_size_;
00070     //! internal fifo_reader_ class acts as signal source
00071     const gnss_shared_ptr<gr::block> fifo_reader_;
00072
00073     gnss_shared_ptr<gr::block> file_sink_;
00074     const bool dump_;
00075     const std::string dump_filename_;
00076 };
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_FIFO_SIGNAL_SOURCE_H
```

13.311 file_signal_source.h File Reference

Interface of a class that reads signals samples from a file and adapts it to a SignalSourceInterface.

```
#include "file_source_base.h"
#include <string>
```

Classes

- class FileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.311.1 Detailed Description

Interface of a class that reads signals samples from a file and adapts it to a SignalSourceInterface.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

This class represents a file signal source. Internally it uses a GNU Radio's gr_file_source as a connector to the data.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file file_signal_source.h.

13.312 file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file file_signal_source.h
00003  * \brief Interface of a class that reads signals samples from a file
00004  * and adapts it to a SignalSourceInterface
00005  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00006  *
00007  * This class represents a file signal source. Internally it uses a GNU Radio's
00008  * gr_file_source as a connector to the data.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_FILE_SIGNAL_SOURCE_H
00022 #define GNSS_SDR_FILE_SIGNAL_SOURCE_H
00023
00024 #include "file_source_base.h"
00025 #include <string>
00026
00027 /** \addtogroup Signal_Source Signal Source
00028  * Classes for Signal Source management.
00029  * \{ */
00030 /** \addtogroup Signal_Source_adapters signal_source_adapters
00031  * Classes that wrap GNU Radio signal sources with a GNSSBlockInterface
00032  * \{ */
00033
00034
00035 class ConfigurationInterface;
00036
00037 /*!
00038  * \brief Class that reads signals samples from a file
00039  * and adapts it to a SignalSourceInterface
00040  */
00041 class FileSignalSource : public FileSourceBase
00042 {
00043 public:
00044     FileSignalSource(ConfigurationInterface const* configuration, std::string const& role,
00045         unsigned int in_streams, unsigned int out_streams,
00046         Concurrent_Queue<pmt::pmt_t>* queue);
00047
00048     ~FileSignalSource() = default;
00049
00050 private:
```

```
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_FILE_SIGNAL_SOURCE_H
```

13.313 file_source_base.h File Reference

Header file of the base class to file-oriented signal_source GNSS blocks.

```
#include "concurrent_queue.h"
#include "sensor_data/sensor_data_source.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/file_source.h>
#include <gnuradio/blocks/throttle.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class FileSourceBase

  Base class to file-oriented SignalSourceBase GNSS blocks.

13.313.1 Detailed Description

Header file of the base class to file-oriented signal_source GNSS blocks.

Author

Jim Melton, 2021. jim.melton(at)sncorp.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file file_source_base.h.

13.314 file_source_base.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file file_source_base.h
00003  * \brief Header file of the base class to file-oriented signal_source GNSS blocks.
00004  * \author Jim Melton, 2021. jim.melton(at)sncorp.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_FILE_SOURCE_BASE_H
00019 #define GNSS_SDR_FILE_SOURCE_BASE_H
00020
00021 #include "concurrent_queue.h"
00022 #include "sensor_data/sensor_data_source.h"
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/file_sink.h>  // for dump
00025 #include <gnuradio/blocks/file_source.h>
00026 #include <gnuradio/blocks/throttle.h>
00027 #include <pmt/pmt.h>
```

```
00028 #include <cstddef>
00029 #include <string>
00030 #include <tuple>
00031
00032
00033 /** \addtogroup Signal_Source
00034  * \{ */
00035 /** \addtogroup Signal_Source_adapters
00036  * \{ */
00037
00038 class ConfigurationInterface;
00039
00040
00041 //! \brief Base class to file-oriented SignalSourceBase GNSS blocks.
00042 //!
00043 //! This class supports the following properties:
00044 //!
00045 //!   .filename - the path to the input file
00046 //!             - may be overridden by the -signal_source or -s command-line arguments
00047 //!
00048 //!   .samples  - number of samples to process (default 0)
00049 //!             - if not specified or 0, read the entire file; otherwise stop after that many samples
00050 //!
00051 //!   .sampling_frequency - the frequency of the sampled data (samples/second)
00052 //!
00053 //!   .item_type - data type of the samples (default "short")
00054 //!
00055 //!   .header_size - the size of a prefixed header to skip in "samples" (default 0)
00056 //!
00057 //!   .seconds_to_skip - number of seconds of lead-in data to skip over (default 0)
00058 //!
00059 //!   .enable_throttle_control - whether to stop reading if the upstream buffer is full (default
      false)
00060 //!
00061 //!   .repeat   - whether to rewind and continue at end of file (default false)
00062 //!
00063 //! (probably abstracted to the base class)
00064 //!
00065 //!   .dump     - whether to archive input data
00066 //!
00067 //!   .dump_filename - if dumping, path to file for output
00068 class FileSourceBase : public SignalSourceBase
00069 {
00070 public:
00071     // Virtual overrides
00072     void connect(gr::top_block_sptr top_block) override;
00073     void disconnect(gr::top_block_sptr top_block) override;
00074     gr::basic_block_sptr get_left_block() override;
00075     gr::basic_block_sptr get_right_block() override;
00076
00077     //! The file to read
00078     std::string filename() const;
00079
00080     //! The item type
00081     std::string item_type() const;
00082
00083     //! The configured size of each item
00084     size_t item_size() override;
00085     virtual size_t item_size() const;  // what the interface **should** have declared
00086
00087     //! Whether to repeat reading after end-of-file
00088     bool repeat() const;
00089
00090     //! The sampling frequency of the source file
00091     int64_t sampling_frequency() const;
00092
00093     //! The number of samples in the file
00094     uint64_t samples() const;
00095
00096 protected:
00097     //! \brief Constructor
00098     //!
00099     //! Subclasses may want to assert default item types that are appropriate to the specific file
00100     //! type supported. Rather than require the item type to be specified in the config file, allow
00101     //! sub-classes to impose their will
00102     FileSourceBase(ConfigurationInterface const* configuration, std::string const& role, std::string
      impl,
00103         Concurrent_Queue<pmt::pmt_t>* queue,
00104         std::string default_item_type = "short");
00105
00106     //! Perform post-construction initialization
00107     void init();
00108
00109     //! Compute the item size, from the item_type(). Subclasses may constrain types that don't make
00110     //  sense. The return of this method is a tuple of item_size and is_complex
00111     virtual std::tuple<size_t, bool> itemTypeToSize();
00112
```

```
00113     //! The number of (possibly unpacked) samples in a (raw) file sample (default=1)
00114     virtual double packetsPerSample() const;
00115
00116     //! Compute the number of samples to skip
00117     virtual uint64_t samplesToSkip() const;
00118
00119     //! Compute the number of samples in the file
00120     uint64_t computeSamplesInFile() const;
00121
00122     //! Abstracted front-end source. Sub-classes may override if they create specialized chains to
00123     //! decode source files into a usable format
00124     virtual gnss_shared_ptr<gr::block> source() const;
00125
00126     //! For complex source chains, the size of the file item may not be the same as the size of the
00127     // "source" (decoded) item. This method allows subclasses to handle these differences
00128     virtual size_t source_item_size() const;
00129     bool is_complex() const;
00130
00131     // Generic access to created objects
00132     gnss_shared_ptr<gr::block> file_source() const;
00133     gnss_shared_ptr<gr::block> valve() const;
00134     gnss_shared_ptr<gr::block> throttle() const;
00135     gnss_shared_ptr<gr::block> sink() const;
00136     SensorDataSource::sptr sensor_data_source() const;
00137
00138     // The methods create the various blocks, if enabled, and return access to them. The created
00139     // object is also held in this class
00140     gr::blocks::file_source::sptr create_file_source();
00141     gr::blocks::throttle::sptr create_throttle();
00142     gnss_shared_ptr<gr::block> create_valve();
00143     gr::blocks::file_sink::sptr create_sink();
00144     SensorDataSource::sptr create_sensor_data_source();
00145
00146     // Subclass hooks to augment created objects, as required
00147     virtual void create_file_source_hook();
00148     virtual void create_throttle_hook();
00149     virtual void create_valve_hook();
00150     virtual void create_sink_hook();
00151
00152     // Subclass hooks for connection/disconnection
00153     virtual void pre_connect_hook(gr::top_block_sptr top_block);
00154     virtual void post_connect_hook(gr::top_block_sptr top_block);
00155     virtual void pre_disconnect_hook(gr::top_block_sptr top_block);
00156     virtual void post_disconnect_hook(gr::top_block_sptr top_block);
00157
00158 private:
00159     gr::blocks::file_source::sptr file_source_;
00160     gr::blocks::throttle::sptr throttle_;
00161     gr::blocks::file_sink::sptr sink_;
00162     SensorDataSource::sptr sensor_data_source_;
00163
00164     // The valve allows only the configured number of samples through, then it closes.
00165
00166     // The framework passes the queue as a naked pointer, rather than a shared pointer, so this
00167     // class has two choices: construct the valve in the ctor, or hold onto the pointer, possibly
00168     // beyond its lifetime. Fortunately, the queue is only used to create the valve, so the
00169     // likelihood of holding a stale pointer is mitigated
00170     gnss_shared_ptr<gr::block> valve_;
00171     Concurrent_Queue<pmt::pmt_t>* queue_;
00172
00173     std::string role_;
00174     std::string filename_;
00175     std::string dump_filename_;
00176     std::string item_type_;
00177     size_t item_size_;
00178     size_t header_size_;  // length (in samples) of the header (if any)
00179     uint64_t samples_;
00180     int64_t sampling_frequency_;  // why is this signed
00181     double minimum_tail_s_;
00182     double seconds_to_skip_;
00183     bool is_complex_;  // a misnomer; if I/Q are interleaved as integer values
00184     bool repeat_;
00185     bool enable_throttle_control_;
00186     bool dump_;
00187
00188     // Configuration for Extra Data source
00189     SensorDataSourceConfiguration sensor_data_source_configuration_;
00190 };
00191
00192 /** \} */
00193 /** \} */
00194 #endif  // GNSS_SDR_FILE_SOURCE_BASE_H
```

13.315 file_timestamp_signal_source.h File Reference

This class reads samples stored in a file and generate stream tags with its timestamp information stored in separated file.

```
#include "configuration_interface.h"
#include "file_source_base.h"
#include "gnss_sdr_timestamp.h"
#include <string>
```

Classes

- class FileTimestampSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.315.1 Detailed Description

This class reads samples stored in a file and generate stream tags with its timestamp information stored in separated file.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file file_timestamp_signal_source.h.

13.316 file_timestamp_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file file_timestamp_signal_source.h
00003  * \brief This class reads samples stored in a file and generate stream tags
00004  * with its timestamp information stored in separated file
00005  * \author Javier Arribas, jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_FILE_TIMESTAMP_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_FILE_TIMESTAMP_SIGNAL_SOURCE_H
00021
00022 #include "configuration_interface.h"
00023 #include "file_source_base.h"
00024 #include "gnss_sdr_timestamp.h"
00025 #include <string>
00026
00027 /** \addtogroup Signal_Source
00028  * \{ */
00029 /** \addtogroup Signal_Source_adapters
00030  * \{ */
00031
00032
00033 /*!
00034  * \brief Class that reads signals samples from a file
00035  * and adapts it to a SignalSourceInterface
00036  */
00037 class FileTimestampSignalSource : public FileSourceBase
00038 {
00039 public:
00040     FileTimestampSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00041         unsigned int in_streams, unsigned int out_streams,
00042         Concurrent_Queue<pmt::pmt_t>* queue);
```

```
00043
00044     ~FileTimestampSignalSource() = default;
00045
00046 protected:
00047     // std::tuple<size_t, bool> itemTypeToSize() override;
00048     // double packetsPerSample() const override;
00049     gnss_shared_ptr<gr::block> source() const override;
00050     void create_file_source_hook() override;
00051     void pre_connect_hook(gr::top_block_sptr top_block) override;
00052     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00053
00054 private:
00055     gnss_shared_ptr<Gnss_Sdr_Timestamp> timestamp_block_;
00056     std::string timestamp_file_;
00057     double timestamp_clock_offset_ms_;
00058 };
00059
00060
00061 /** \} */
00062 /** \} */
00063 #endif  // GNSS_SDR_FILE_TIMESTAMP_SIGNAL_SOURCE_H
```

13.317 flexiband_signal_source.h File Reference

ignal Source adapter for the Teleorbit Flexiband front-end device. This adapter requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR)

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/char_to_float.h>
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/float_to_complex.h>
#include <gnuradio/blocks/null_sink.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class FlexibandSignalSource

  This class configures and reads samples from Teleorbit Flexiband front-end. This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR).

13.317.1 Detailed Description

ignal Source adapter for the Teleorbit Flexiband front-end device. This adapter requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR)

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file flexiband_signal_source.h.

13.318 flexiband_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file flexiband_signal_source.h
00003  * \brief ignal Source adapter for the Teleorbit Flexiband front-end device.
00004  * This adapter requires a Flexiband GNU Radio driver
00005  * installed (not included with GNSS-SDR)
00006  * \author Javier Arribas, jarribas(at)cttc.es
```

```
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_FLEXIBAND_SIGNAL_SOURCE_H
00021 #define GNSS_SDR_FLEXIBAND_SIGNAL_SOURCE_H
00022
00023 #include "concurrent_queue.h"
00024 #include "signal_source_base.h"
00025 #include <gnuradio/blocks/char_to_float.h>
00026 #include <gnuradio/blocks/file_sink.h>
00027 #include <gnuradio/blocks/float_to_complex.h>
00028 #include <gnuradio/blocks/null_sink.h>
00029 #include <gnuradio/hier_block2.h>
00030 #include <pmt/pmt.h>
00031 #include <memory>
00032 #include <string>
00033 #include <vector>
00034
00035
00036 /** \addtogroup Signal_Source
00037  * \{ */
00038 /** \addtogroup Signal_Source_adapters
00039  * \{ */
00040
00041
00042 class ConfigurationInterface;
00043
00044 /*!
00045  * \brief This class configures and reads samples from Teleorbit Flexiband front-end.
00046  * This software requires a Flexiband GNU Radio driver installed (not included with GNSS-SDR).
00047  */
00048 class FlexibandSignalSource : public SignalSourceBase
00049 {
00050 public:
00051     FlexibandSignalSource(const ConfigurationInterface* configuration,
00052         const std::string& role, unsigned int in_stream,
00053         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00054
00055     ~FlexibandSignalSource() = default;
00056
00057     inline size_t item_size() override
00058     {
00059         return item_size_;
00060     }
00061
00062     void connect(gr::top_block_sptr top_block) override;
00063     void disconnect(gr::top_block_sptr top_block) override;
00064     gr::basic_block_sptr get_left_block() override;
00065     gr::basic_block_sptr get_right_block() override;
00066     gr::basic_block_sptr get_right_block(int RF_channel) override;
00067
00068 private:
00069     gnss_shared_ptr<gr::block> flexiband_source_;
00070
00071     std::vector<gnss_shared_ptr<gr::block» char_to_float;
00072     std::vector<gnss_shared_ptr<gr::block» float_to_complex_;
00073     std::vector<gr::blocks::null_sink::sptr> null_sinks_;
00074
00075     std::string item_type_;
00076     std::string firmware_filename_;
00077     std::string signal_file;
00078
00079     size_t item_size_;
00080     unsigned int in_stream_;
00081     unsigned int out_stream_;
00082
00083     int gain1_;
00084     int gain2_;
00085     int gain3_;
00086     int usb_packet_buffer_size_;
00087     int n_channels_;
00088     int sel_ch_;
00089
00090     bool AGC_;
00091     bool flag_read_file;
00092 };
00093
```

```
00094
00095 /** \} */
00096 /** \} */
00097 #endif  // GNSS_SDR_FLEXIBAND_SIGNAL_SOURCE_H
```

13.319 fmcomms2_signal_source.h File Reference

Interface to use SDR hardware based in FMCOMMS2 driver from analog devices, for example FMCOMMS4 and ADALM-PLUTO (PlutoSdr).

```
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <iio/fmcomms2_source.h>
#include "concurrent_queue.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
```

Classes

- class Fmcomms2SignalSource

13.319.1 Detailed Description

Interface to use SDR hardware based in FMCOMMS2 driver from analog devices, for example FMCOMMS4 and ADALM-PLUTO (PlutoSdr).

Author

Rodrigo Muñoz, 2017. rmunozl(at)inacap.cl, rodrigo.munoz(at)proteinlab.cl

13.319.1.1 This class represent a fmcomms2 signal source. It use the gr_iio block

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fmcomms2_signal_source.h.

13.320 fmcomms2_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fmcomms2_signal_source.h
00003  * \brief Interface to use SDR hardware based in FMCOMMS2 driver from analog
00004  * devices, for example FMCOMMS4 and ADALM-PLUTO (PlutoSdr)
00005  * \author Rodrigo Muñoz, 2017. rmunozl(at)inacap.cl, rodrigo.munoz(at)proteinlab.cl
00006  *
00007  *
00008  * This class represent a fmcomms2 signal source. It use the gr_iio block
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_FMCOMMS2_SIGNAL_SOURCE_H
00021 #define GNSS_SDR_FMCOMMS2_SIGNAL_SOURCE_H
00022
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/file_sink.h>
00025 #if GRIIO_INCLUDE_HAS_GNURADIO
00026 #include <gnuradio/iio/fmcomms2_source.h>
00027 #else
00028 #include <iio/fmcomms2_source.h>
00029 #endif
```

```
00030 #include "concurrent_queue.h"
00031 #include <pmt/pmt.h>
00032 #include <cstdint>
00033 #include <string>
00034
00035
00036 /** \addtogroup Signal_Source
00037  * \{ */
00038 /** \addtogroup Signal_Source_adapters
00039  * \{ */
00040
00041
00042 class ConfigurationInterface;
00043
00044 class Fmcomms2SignalSource : public SignalSourceBase
00045 {
00046 public:
00047     Fmcomms2SignalSource(const ConfigurationInterface* configuration,
00048         const std::string& role, unsigned int in_stream,
00049         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00050
00051     ~Fmcomms2SignalSource();
00052
00053     inline size_t item_size() override
00054     {
00055         return item_size_;
00056     }
00057
00058     void connect(gr::top_block_sptr top_block) override;
00059     void disconnect(gr::top_block_sptr top_block) override;
00060     gr::basic_block_sptr get_left_block() override;
00061     gr::basic_block_sptr get_right_block() override;
00062
00063 private:
00064     const std::string default_gain_mode = std::string("slow_attack");
00065     const double default_tx_attenuation_db = -10.0;
00066
00067 #if GNURADIO_API_IIO
00068 #if GR_IIO_TEMPLATIZED_API
00069     gr::iio::fmcomms2_source<gr_complex>::sptr fmcomms2_source_f32c_;
00070 #else
00071     gr::iio::fmcomms2_source::sptr fmcomms2_source_f32c_;
00072 #endif
00073 #else
00074     gr::iio::fmcomms2_source_f32c::sptr fmcomms2_source_f32c_;
00075 #endif
00076     gnss_shared_ptr<gr::block> valve_;
00077     gr::blocks::file_sink::sptr file_sink_;
00078
00079     std::string item_type_;
00080     std::string dump_filename_;
00081
00082     // Front-end settings
00083     std::string uri_;  // device direction
00084     std::string gain_mode_rx1_;
00085     std::string gain_mode_rx2_;
00086     std::string rf_port_select_;
00087     std::string filter_file_;
00088     std::string filter_source_;
00089     std::string filter_filename_;
00090
00091     int64_t samples_;
00092     size_t item_size_;
00093
00094     double rf_gain_rx1_;
00095     double rf_gain_rx2_;
00096     uint64_t freq_;  // frequency of local oscillator
00097     uint64_t sample_rate_;
00098     uint64_t bandwidth_;
00099     uint64_t buffer_size_;  // reception buffer
00100     float Fpass_;
00101     float Fstop_;
00102     unsigned int in_stream_;
00103     unsigned int out_stream_;
00104     int RF_channels_;
00105
00106     // DDS configuration for LO generation for external mixer
00107     double scale_dds_dbfs_;
00108     double phase_dds_deg_;
00109     double tx_attenuation_db_;
00110     uint64_t freq_dds_tx_hz_;
00111     uint64_t freq_rf_tx_hz_;
00112     uint64_t tx_bandwidth_;
00113     bool enable_dds_lo_;
00114
00115     bool rx1_en_;
00116     bool rx2_en_;
```

```
00117     bool quadrature_;
00118     bool rf_dc_;
00119     bool bb_dc_;
00120     bool filter_auto_;
00121     bool rf_shutdown_;
00122     bool dump_;
00123 };
00124
00125
00126 /** \} */
00127 /** \} */
00128 #endif  // GNSS_SDR_FMCOMMS2_SIGNAL_SOURCE_H
```

13.321 fmcomms5_signal_source_fpga.h File Reference

Signal source for the Analog Devices FMCOMMS5 directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file.

```
#include "concurrent_queue.h"
#include "fpga_buffer_monitor.h"
#include "fpga_dynamic_bit_selection.h"
#include "fpga_switch.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
#include <thread>
```

Classes

- class Fmcomms5SignalSourceFPGA

13.321.1 Detailed Description

Signal source for the Analog Devices FMCOMMS5 directly connected to the FPGA accelerators. This source implements only the AD9361 control. It is NOT compatible with conventional SDR acquisition and tracking blocks. Please use the fmcomms2 source if conventional SDR acquisition and tracking is selected in the configuration file.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file fmcomms5_signal_source_fpga.h.

13.322 fmcomms5_signal_source_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fmcomms5_signal_source_fpga.h
00003  * \brief Signal source for the Analog Devices FMCOMMS5 directly connected
00004  * to the FPGA accelerators.
00005  * This source implements only the AD9361 control. It is NOT compatible with
00006  * conventional SDR acquisition and tracking blocks.
00007  * Please use the fmcomms2 source if conventional SDR acquisition and tracking
00008  * is selected in the configuration file.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
```

```
00020
00021 #ifndef GNSS_SDR_FMCOMMS5_SIGNAL_SOURCE_FPGA_H
00022 #define GNSS_SDR_FMCOMMS5_SIGNAL_SOURCE_FPGA_H
00023
00024 #include "concurrent_queue.h"
00025 #include "fpga_buffer_monitor.h"
00026 #include "fpga_dynamic_bit_selection.h"
00027 #include "fpga_switch.h"
00028 #include "gnss_block_interface.h"
00029 #include "signal_source_base.h"
00030 #include <pmt/pmt.h>
00031 #include <cstdint>
00032 #include <memory>
00033 #include <mutex>
00034 #include <string>
00035 #include <thread>
00036
00037
00038 /** \addtogroup Signal_Source
00039  * \{ */
00040 /** \addtogroup Signal_Source_adapters
00041  * \{ */
00042
00043
00044 class ConfigurationInterface;
00045
00046 class Fmcomms5SignalSourceFPGA : public SignalSourceBase
00047 {
00048 public:
00049     Fmcomms5SignalSourceFPGA(const ConfigurationInterface *configuration,
00050         const std::string &role, unsigned int in_stream,
00051         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t> *queue);
00052
00053     ~Fmcomms5SignalSourceFPGA();
00054
00055     inline size_t item_size() override
00056     {
00057         return item_size_;
00058     }
00059
00060     void connect(gr::top_block_sptr top_block) override;
00061     void disconnect(gr::top_block_sptr top_block) override;
00062     gr::basic_block_sptr get_left_block() override;
00063     gr::basic_block_sptr get_right_block() override;
00064
00065 private:
00066     const std::string DEFAULT_NUM_FILENAME = std::string("FPGA_buffer_monitor_dump.dat");
00067     const std::string DEFAULT_RF_PORT_SELECT = std::string("A_BALANCED");
00068     const std::string DEFAULT_GAIN_MODE = std::string("slow_attack");
00069     const double DEFAULT_MANUAL_GAIN_RX1 = 64.0;
00070     const double DEFAULT_MANUAL_GAIN_RX2 = 64.0;
00071     const uint64_t DEFAULT_BANDWIDTH = 12500000;
00072
00073     // perform dynamic bit selection every 500 ms by default
00074     const uint32_t GAIN_CONTROL_PERIOD_ms = 500;
00075     // check buffer overflow and perform buffer monitoring every 1s by default
00076     const uint32_t BUFFER_MONITOR_PERIOD_ms = 1000;
00077     // buffer overflow and buffer monitoring initial delay
00078     const uint32_t BUFFER_MONITOR_INITIAL_DELAY_ms = 2000;
00079     const int32_t REAL_TIME_MODE = 2;
00080
00081     void run_dynamic_bit_selection_process();
00082     void run_buffer_monitor_process();
00083
00084     mutable std::mutex dynamic_bit_selection_mutex;
00085     mutable std::mutex buffer_monitor_mutex;
00086
00087     std::thread thread_dynamic_bit_selection;
00088     std::thread thread_buffer_monitor;
00089
00090     std::shared_ptr<Fpga_Switch> switch_fpga;
00091     std::shared_ptr<Fpga_dynamic_bit_selection> dynamic_bit_selection_fpga;
00092     std::shared_ptr<Fpga_buffer_monitor> buffer_monitor_fpga;
00093
00094     std::string gain_mode_rx1_;
00095     std::string gain_mode_rx2_;
00096     std::string rf_port_select_;
00097     std::string filter_file_;
00098     std::string filter_source_;
00099     std::string filter_filename_;
00100
00101     double rf_gain_rx1_;
00102     double rf_gain_rx2_;
00103
00104     uint64_t freq0_;  // frequency of local oscillator for ADRV9361-A
00105     uint64_t freq1_;  // frequency of local oscillator for ADRV9361-B
00106     uint64_t sample_rate_;
```

```
00107     uint64_t bandwidth_;
00108
00109     float Fpass_;
00110     float Fstop_;
00111     uint32_t in_stream_;
00112     uint32_t out_stream_;
00113
00114     size_t item_size_;
00115
00116     bool quadrature_;
00117     bool rf_dc_;
00118     bool bb_dc_;
00119     bool rx1_enable_;
00120     bool rx2_enable_;
00121     bool enable_dynamic_bit_selection_;
00122     bool enable_ovf_check_buffer_monitor_active_;
00123     bool dump_;
00124     bool rf_shutdown_;
00125 };
00126
00127
00128 /** \} */
00129 /** \} */
00130 #endif  // GNSS_SDR_FMCOMMS5_SIGNAL_SOURCE_FPGA_H
```

13.323 four_bit_cpx_file_signal_source.h File Reference

Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface.

```
#include "file_source_base.h"
#include "gnss_sdr_timestamp.h"
#include "unpack_byte_4bit_samples.h"
#include <gnuradio/blocks/interleaved_short_to_complex.h>
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class FourBitCpxFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.323.1 Detailed Description

Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface.

Author

Javier Arribas, 2015 jarribas(at)cttc.es

This class represents a file signal source.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file four_bit_cpx_file_signal_source.h.

13.324 four_bit_cpx_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file four_bit_cpx_file_signal_source.h
00003  * \brief Interface of a class that reads signals samples from a 2 bit complex sampler front-end file
00004  * and adapts it to a SignalSourceInterface.
00005  * \author Javier Arribas, 2015 jarribas(at)cttc.es
00006  *
00007  * This class represents a file signal source.
00008  *
00009  * -----------------------------------------------------------------------------
```

```
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_FOUR_BIT_CPX_FILE_SIGNAL_SOURCE_H
00021 #define GNSS_SDR_FOUR_BIT_CPX_FILE_SIGNAL_SOURCE_H
00022
00023 #include "file_source_base.h"
00024 #include "gnss_sdr_timestamp.h"
00025 #include "unpack_byte_4bit_samples.h"
00026 #include <gnuradio/blocks/interleaved_short_to_complex.h>
00027 #include <cstddef>
00028 #include <string>
00029 #include <tuple>
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 /*!
00040  * \brief Class that reads signals samples from a file
00041  * and adapts it to a SignalSourceInterface
00042  */
00043 class FourBitCpxFileSignalSource : public FileSourceBase
00044 {
00045 public:
00046     FourBitCpxFileSignalSource(const ConfigurationInterface* configuration,
00047         const std::string& role,
00048         unsigned int in_streams,
00049         unsigned int out_streams,
00050         Concurrent_Queue<pmt::pmt_t>* queue);
00051
00052     ~FourBitCpxFileSignalSource() = default;
00053
00054 protected:
00055     std::tuple<size_t, bool> itemTypeToSize() override;
00056     double packetsPerSample() const override;
00057     gnss_shared_ptr<gr::block> source() const override;
00058     void create_file_source_hook() override;
00059     void pre_connect_hook(gr::top_block_sptr top_block) override;
00060     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00061
00062 private:
00063     unpack_byte_4bit_samples_sptr unpack_byte_;
00064     gr::blocks::interleaved_short_to_complex::sptr inter_shorts_to_cpx_;
00065     gnss_shared_ptr<Gnss_Sdr_Timestamp> timestamp_block_;
00066     std::string sample_type_;
00067     std::string timestamp_file_;
00068     double timestamp_clock_offset_ms_;
00069     bool reverse_interleaving_;
00070 };
00071
00072
00073 /** \} */
00074 /** \} */
00075 #endif  // GNSS_SDR_FOUR_BIT_CPX_FILE_SIGNAL_SOURCE_H
```

13.325 gen_signal_source.h File Reference

It wraps blocks that generates synthesized GNSS signal and filters it.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "signal_source_interface.h"
#include <pmt/pmt.h>
#include <memory>
#include <string>
```

Classes

- class GenSignalSource

  This class wraps blocks that generates synthesized GNSS signal and filters the signal.

13.325.1 Detailed Description

It wraps blocks that generates synthesized GNSS signal and filters it.

Author

Marc Molina, 2013. marc.molina.pena@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gen_signal_source.h.

13.326 gen_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gen_signal_source.h
00003  * \brief It wraps blocks that generates synthesized GNSS signal and filters
00004  *  it.
00005  * \author Marc Molina, 2013. marc.molina.pena@gmail.com
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GEN_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_GEN_SIGNAL_SOURCE_H
00021
00022
00023 #include "concurrent_queue.h"
00024 #include "gnss_block_interface.h"
00025 #include "signal_source_interface.h"
00026 #include <pmt/pmt.h>
00027 #include <memory>
00028 #include <string>
00029
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036
00037 /*!
00038  * \brief This class wraps blocks that generates synthesized GNSS signal and
00039  * filters the signal.
00040  */
00041 class GenSignalSource : public SignalSourceInterface
00042 {
00043 public:
00044     //! Constructor
00045     GenSignalSource(std::shared_ptr<GNSSBlockInterface> signal_generator,
      std::shared_ptr<GNSSBlockInterface> filter,
00046         std::string role, Concurrent_Queue<pmt::pmt_t> *queue);
00047
00048     void connect(gr::top_block_sptr top_block) override;
00049     void disconnect(gr::top_block_sptr top_block) override;
00050     gr::basic_block_sptr get_left_block() override;
00051     gr::basic_block_sptr get_right_block() override;
00052
00053     inline std::string role() override { return role_; }
00054     //! Returns "Signal Source"
00055     inline std::string implementation() override { return "Signal Source"; }
00056     inline size_t item_size() override { return 0; }
00057     inline size_t getRfChannels() const final { return 0; }
00058
```

```
00059     inline std::shared_ptr<GNSSBlockInterface> signal_generator() const { return signal_generator_; }
00060
00061 private:
00062     std::shared_ptr<GNSSBlockInterface> signal_generator_;
00063     std::shared_ptr<GNSSBlockInterface> filter_;
00064     std::string role_;
00065     std::string implementation_;
00066     bool connected_;
00067 };
00068
00069
00070 /** \} */
00071 /** \} */
00072 #endif  // GNSS_SDR_GEN_SIGNAL_SOURCE_H
```

13.327 gnmax_signal_source.h File Reference

gnMAX2769 USB dongle GPS RF front-end signal sampler driver

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <stdexcept>
#include <string>
```

Classes

- class GnMaxSignalSource

  This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler.

13.327.1 Detailed Description

gnMAX2769 USB dongle GPS RF front-end signal sampler driver

Author

Wojciech Kazubski, wk(at)ire.pw.edu.pl

Javier Arribas, jarribas(at)cttc.es

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors)
GNSS-SDR is a software defined Global Navigation Satellite Systems receiver
This file is part of GNSS-SDR.
SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnmax_signal_source.h.

13.328 gnmax_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnmax_signal_source.h
00003  * \brief gnMAX2769 USB dongle GPS RF front-end signal sampler driver
00004  * \author Wojciech Kazubski, wk(at)ire.pw.edu.pl
00005  * \author Javier Arribas, jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00010  *
00011  * GNSS-SDR is a software defined Global Navigation
00012  *          Satellite Systems receiver
00013  *
00014  * This file is part of GNSS-SDR.
00015  *
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
```

```
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_GNMAX_SIGNAL_SOURCE_H_
00022 #define GNSS_SDR_GNMAX_SIGNAL_SOURCE_H_
00023
00024 #include "concurrent_queue.h"
00025 #include "signal_source_base.h"
00026 #include <gnuradio/blocks/file_sink.h>
00027 #include <pmt/pmt.h>
00028 #include <cstdint>
00029 #include <memory>
00030 #include <stdexcept>
00031 #include <string>
00032
00033 /** \addtogroup Signal_Source
00034  * \{ */
00035 /** \addtogroup Signal_Source_adapters
00036  * \{ */
00037
00038
00039 class ConfigurationInterface;
00040
00041 /*!
00042  * \brief This class reads samples from a gnMAX2769 USB dongle, a RF front-end signal sampler
00043  */
00044 class GnMaxSignalSource : public SignalSourceBase
00045 {
00046 public:
00047     GnMaxSignalSource(const ConfigurationInterface* configuration,
00048         const std::string& role, unsigned int in_stream,
00049         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00050
00051     virtual ~GnMaxSignalSource();
00052
00053     inline size_t item_size() override
00054     {
00055         return item_size_;
00056     }
00057
00058     void connect(gr::top_block_sptr top_block) override;
00059     void disconnect(gr::top_block_sptr top_block) override;
00060     gr::basic_block_sptr get_left_block() override;
00061     gr::basic_block_sptr get_right_block() override;
00062
00063 private:
00064     gr::block_sptr gnmax_source_;
00065     gnss_shared_ptr<gr::block> valve_;
00066     gr::blocks::file_sink::sptr file_sink_;
00067
00068     std::string item_type_;
00069     std::string dump_filename_;
00070
00071     // Front-end settings
00072     bool bias_;
00073     int bias__;
00074     int ant_;
00075     unsigned long freq_;  // frequency of local oscilator
00076     float freq__;
00077     unsigned long bw_;
00078     int bw__;
00079     bool zeroif_;
00080     int zeroif__;
00081
00082     size_t item_size_;
00083     int64_t samples_;
00084
00085     unsigned int in_stream_;
00086     unsigned int out_stream_;
00087
00088     bool dump_;
00089 };
00090
00091
00092 /** \} */
00093 /** \} */
00094 #endif /*GNSS_SDR_GNMAX_SIGNAL_SOURCE_H_*/
```

13.329 gss6450_file_signal_source.h File Reference

Implementation of a class that reads GSS6450 sample files and adapts it to a SignalSourceInterface.

```
#include "concurrent_queue.h"
#include "gnss_sdr_valve.h"
```

```
#include "signal_source_base.h"
#include "unpack_gss6450_samples.h"
#include <gnuradio/blocks/endian_swap.h>
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/file_source.h>
#include <gnuradio/blocks/null_sink.h>
#include <gnuradio/blocks/stream_to_streams.h>
#include <gnuradio/blocks/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class GSS6450FileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.329.1 Detailed Description

Implementation of a class that reads GSS6450 sample files and adapts it to a SignalSourceInterface.

Author

Antonio Ramos, 2017 antonio.ramos(at)cttc.es

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gss6450_file_signal_source.h.

13.330 gss6450_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gss6450_file_signal_source.h
00003  * \brief Implementation of a class that reads GSS6450 sample files
00004  * and adapts it to a SignalSourceInterface.
00005  * \author Antonio Ramos, 2017 antonio.ramos(at)cttc.es
00006  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GSS6450_FILE_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_GSS6450_FILE_SIGNAL_SOURCE_H
00021
00022 #include "concurrent_queue.h"
00023 #include "gnss_sdr_valve.h"
00024 #include "signal_source_base.h"
00025 #include "unpack_gss6450_samples.h"
00026 #include <gnuradio/blocks/endian_swap.h>
00027 #include <gnuradio/blocks/file_sink.h>
00028 #include <gnuradio/blocks/file_source.h>
00029 #include <gnuradio/blocks/null_sink.h>
00030 #include <gnuradio/blocks/stream_to_streams.h>
00031 #include <gnuradio/blocks/throttle.h>
00032 #include <gnuradio/hier_block2.h>
```

```
00033 #include <pmt/pmt.h>
00034 #include <cstddef>
00035 #include <cstdint>
00036 #include <string>
00037 #include <vector>
00038
00039
00040 /** \addtogroup Signal_Source
00041  * \{ */
00042 /** \addtogroup Signal_Source_adapters
00043  * \{ */
00044
00045
00046 class ConfigurationInterface;
00047
00048 /*!
00049  * \brief Class that reads signals samples from a file
00050  * and adapts it to a SignalSourceInterface
00051  */
00052 class GSS6450FileSignalSource : public SignalSourceBase
00053 {
00054 public:
00055     GSS6450FileSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00056         uint32_t in_streams, uint32_t out_streams, Concurrent_Queue<pmt::pmt_t>* queue);
00057
00058     inline size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block(int RF_channel) override;
00067     gr::basic_block_sptr get_right_block() override;
00068     size_t getRfChannels() const override;
00069
00070     inline std::string filename() const
00071     {
00072         return filename_;
00073     }
00074
00075     inline std::string item_type() const
00076     {
00077         return item_type_;
00078     }
00079
00080     inline bool repeat() const
00081     {
00082         return repeat_;
00083     }
00084
00085     inline int64_t sampling_frequency() const
00086     {
00087         return sampling_frequency_;
00088     }
00089
00090     inline uint64_t samples() const
00091     {
00092         return samples_;
00093     }
00094
00095 private:
00096     struct Gss6450FileLayout
00097     {
00098         int64_t data_offset = -1;
00099         int32_t channels = 0;
00100         uint32_t adc_bits = 0;
00101         bool gss6425_compatible = false;
00102     };
00103
00104     static uint32_t read_le_word(const std::vector<uint8_t>& bytes, size_t offset);
00105     static bool matches_sync_channel(uint32_t word, uint8_t channel_pattern);
00106     static uint32_t bits_from_header(const std::string& header, const std::string& filename);
00107     static bool is_gss6425_compatibility_header(const std::string& header);
00108     static Gss6450FileLayout detect_gss6450_layout(const std::string& filename);
00109     static bool is_valid_gss6450_bits(uint32_t bits);
00110     static bool is_ascii_digit(char value);
00111     static bool is_ascii_space(char value);
00112     static char to_lower_ascii(char value);
00113     static size_t find_case_insensitive(const std::string& text, const std::string& pattern, size_t
      start_pos = 0);
00114
00115     gr::blocks::file_source::sptr file_source_;
00116     gr::blocks::stream_to_streams::sptr stream_to_streams_;
00117     std::vector<gnss_shared_ptr<gr::block» valve_vec_;
00118     std::vector<gr::blocks::endian_swap::sptr> endian_vec_;
```

```
00119     std::vector<gr::blocks::null_sink::sptr> null_sinks_;
00120     std::vector<unpack_gss6450_samples_sptr> unpack_gss6450_vec_;
00121     std::vector<gr::blocks::file_sink::sptr> sink_vec_;
00122     std::vector<gr::blocks::throttle::sptr> throttle_vec_;
00123     std::string filename_;
00124     std::string dump_filename_;
00125     std::string item_type_;
00126     uint64_t samples_;
00127     int64_t bytes_to_skip_;
00128     int64_t sampling_frequency_;
00129     size_t item_size_;
00130     uint32_t in_streams_;
00131     uint32_t out_streams_;
00132     uint32_t adc_bits_;
00133     int32_t n_channels_;
00134     int32_t sel_ch_;
00135     size_t rf_channels_;
00136     bool gss6425_compatible_;
00137     bool repeat_;
00138     bool dump_;  // Enables dumping the gr_complex sample output
00139     bool enable_throttle_control_;
00140     bool endian_swap_;
00141 };
00142
00143
00144 /** \} */
00145 /** \} */
00146 #endif  // GNSS_SDR_GSS6450_FILE_SIGNAL_SOURCE_H
```

13.331 ion_gsms_signal_source.h File Reference

GNSS-SDR Signal Source that reads sample streams following ION's GNSS-SDR metadata standard.

```
#include "configuration_interface.h"
#include "file_source_base.h"
#include "gnss_sdr_filesystem.h"
#include "gnss_sdr_timestamp.h"
#include "ion_gsms.h"
#include <GnssMetadata.h>
#include <cstddef>
#include <cstdint>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class IONGSMSSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.331.1 Detailed Description

GNSS-SDR Signal Source that reads sample streams following ION's GNSS-SDR metadata standard.

Author

Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ion_gsms_signal_source.h.

13.332 ion_gsms_signal_source.h

Go to the documentation of this file.

```
00001 /*!
```

```
00002  * \file ion_gsms_signal_source.h
00003  * \brief GNSS-SDR Signal Source that reads sample streams following ION's GNSS-SDR metadata standard
00004  * \author Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com
00005  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_ION_GSMS_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_ION_GSMS_SIGNAL_SOURCE_H
00021
00022 #include "configuration_interface.h"
00023 #include "file_source_base.h"
00024 #include "gnss_sdr_filesystem.h"
00025 #include "gnss_sdr_timestamp.h"
00026 #include "ion_gsms.h"
00027 #include <GnssMetadata.h>
00028 #include <cstddef>
00029 #include <cstdint>
00030 #include <memory>
00031 #include <string>
00032 #include <vector>
00033
00034 /** \addtogroup Signal_Source
00035  * \{ */
00036 /** \addtogroup Signal_Source_adapters
00037  * \{ */
00038
00039 /*!
00040  * \brief Class that reads signals samples from a file
00041  * and adapts it to a SignalSourceInterface
00042  */
00043 class IONGSMSSignalSource : public SignalSourceBase
00044 {
00045 public:
00046     IONGSMSSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00047         unsigned int in_streams, unsigned int out_streams,
00048         Concurrent_Queue<pmt::pmt_t>* queue);
00049
00050     ~IONGSMSSignalSource() override = default;
00051
00052 protected:
00053     void connect(gr::top_block_sptr top_block) override;
00054     void disconnect(gr::top_block_sptr top_block) override;
00055
00056     gr::basic_block_sptr get_left_block() override;
00057     gr::basic_block_sptr get_right_block() override;
00058     gr::basic_block_sptr get_right_block(int RF_channel) override;
00059     size_t getRfChannels() const override;
00060
00061     inline size_t item_size() override
00062     {
00063         if (sources_.empty())
00064             {
00065                 return 0;
00066             }
00067         return sources_.begin()->source->output_stream_item_size(0);
00068     }
00069
00070 private:
00071     static constexpr double kMinimumTailSeconds = 0.2;
00072
00073     struct StreamSourceData
00074     {
00075         IONGSMSFileSource::sptr source;
00076         std::int64_t sampling_frequency = 0;
00077     };
00078
00079     struct MetadataFileData
00080     {
00081         std::size_t sequence = 0;
00082         bool has_file = false;
00083         GnssMetadata::File file;
00084         fs::path metadata_directory;
00085     };
00086
00087     static std::vector<std::string> parse_comma_list(const std::string& str);
00088     static bool block_contains_stream(const GnssMetadata::Block& block, const
```

```
      std::vector<std::string>& stream_ids);
00089     static std::size_t chunk_cycle_bytes(const GnssMetadata::Block& block);
00090     static std::size_t infer_block_cycles(
00091         const fs::path& data_filepath,
00092         const GnssMetadata::Block& block,
00093         std::size_t block_start_offset,
00094         bool block_extends_to_eof);
00095     static std::size_t block_storage_bytes(
00096         const fs::path& data_filepath,
00097         const GnssMetadata::Block& block,
00098         std::size_t block_start_offset,
00099         bool block_extends_to_eof);
00100     static const char* file_uri_scheme();
00101     static bool starts_with(const std::string& value, const std::string& prefix);
00102     static fs::path absolute_path_key(const fs::path& path);
00103     static fs::path resolve_local_metadata_uri(const fs::path& metadata_path, const std::string& uri);
00104
00105     std::vector<StreamSourceData> make_stream_sources(const std::vector<std::string>& stream_ids)
      const;
00106     std::vector<MetadataFileData> ordered_metadata_files() const;
00107     const GnssMetadata::System* resolve_system(const GnssMetadata::System& system) const;
00108     double lane_base_frequency_hz(const GnssMetadata::Lane& lane) const;
00109     std::int64_t stream_sampling_frequency_hz(
00110         const GnssMetadata::Lane& lane,
00111         const GnssMetadata::Block& block,
00112         const std::string& stream_id) const;
00113     static std::int64_t reconcile_sampling_frequency(
00114         std::int64_t current_frequency,
00115         std::int64_t candidate_frequency,
00116         const std::string& stream_id);
00117
00118     void load_metadata();
00119     void load_metadata_file(
00120         const fs::path& metadata_path,
00121         GnssMetadata::Metadata& metadata,
00122         std::vector<std::string>& include_stack);
00123     std::uint64_t valve_sample_count(std::uint64_t total_sample_count, std::int64_t
      sampling_frequency) const;
00124
00125     std::vector<std::string> stream_ids_;
00126     std::vector<StreamSourceData> sources_;
00127     std::vector<gnss_shared_ptr<gr::block» copy_blocks_;
00128     std::vector<gnss_shared_ptr<gr::block» valves_;
00129
00130     std::string metadata_filepath_;
00131     std::shared_ptr<GnssMetadata::Metadata> metadata_;
00132     std::vector<MetadataFileData> metadata_files_;
00133
00134     gnss_shared_ptr<Gnss_Sdr_Timestamp> timestamp_block_;
00135     std::string timestamp_file_;
00136     double minimum_tail_s_;
00137     int64_t sampling_frequency_;
00138
00139     uint32_t in_streams_;
00140     uint32_t out_streams_;
00141 };
00142
00143
00144 /** \} */
00145 /** \} */
00146 #endif  // GNSS_SDR_ION_GSMS_SIGNAL_SOURCE_H
```

13.333 labsat_signal_source.h File Reference

LabSat version 2, 3, and 3 Wideband format reader.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class LabsatSignalSource

  This class reads samples stored in LabSat version 2, 3, and 3 Wideband format.

13.333.1 Detailed Description

LabSat version 2, 3, and 3 Wideband format reader.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file labsat_signal_source.h.

13.334 labsat_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file labsat_signal_source.h
00003  * \brief LabSat version 2, 3, and 3 Wideband format reader
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_LABSAT_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_LABSAT_SIGNAL_SOURCE_H
00020
00021 #include "concurrent_queue.h"
00022 #include "gnss_block_interface.h"
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/file_sink.h>
00025 #include <gnuradio/blocks/throttle.h>
00026 #include <gnuradio/hier_block2.h>
00027 #include <pmt/pmt.h>
00028 #include <memory>
00029 #include <string>
00030 #include <vector>
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_adapters
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class reads samples stored in LabSat version 2, 3, and 3 Wideband
00042  * format.
00043  */
00044 class LabsatSignalSource : public SignalSourceBase
00045 {
00046 public:
00047     LabsatSignalSource(const ConfigurationInterface* configuration,
00048         const std::string& role, unsigned int in_stream,
00049         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00050
00051     ~LabsatSignalSource() = default;
00052
00053     inline size_t item_size() override
00054     {
00055         return item_size_;
00056     }
00057
00058     size_t getRfChannels() const override;
00059     void connect(gr::top_block_sptr top_block) override;
```

```
00060     void disconnect(gr::top_block_sptr top_block) override;
00061     gr::basic_block_sptr get_left_block() override;
00062     gr::basic_block_sptr get_right_block() override;
00063     gr::basic_block_sptr get_right_block(int i) override;
00064
00065 private:
00066     gr::block_sptr labsat23_source_;
00067     std::vector<gr::blocks::file_sink::sptr> file_sink_;
00068     std::vector<gr::blocks::throttle::sptr> throttle_;
00069     std::vector<int> channels_selector_vec_;
00070
00071     std::string item_type_;
00072     std::string filename_;
00073     std::string dump_filename_;
00074
00075     size_t item_size_;
00076
00077     unsigned int in_stream_;
00078     unsigned int out_stream_;
00079
00080     bool enable_throttle_control_;
00081     bool dump_;
00082 };
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_LABSAT_SIGNAL_SOURCE_H
```

13.335 limesdr_signal_source.h

```
00001 /*!
00002  * \file limesdr_signal_source.cc
00003  * \brief Signal source for LimeSDR front-end
00004  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_LIMESDR_SIGNAL_SOURCE_H
00018 #define GNSS_SDR_LIMESDR_SIGNAL_SOURCE_H
00019
00020 #include "concurrent_queue.h"
00021 #include "signal_source_base.h"
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <pmt/pmt.h>
00024 #include <cstdint>
00025 #include <limesdr/source.h>
00026 #include <memory>
00027 #include <stdexcept>
00028 #include <string>
00029
00030 /** \addtogroup Signal_Source
00031  * \{ */
00032 /** \addtogroup Signal_Source_adapters
00033  * \{ */
00034
00035
00036 class ConfigurationInterface;
00037
00038 /*!
00039  * \brief This class instantiates the LimeSDR gnuradio signal source.
00040  * It has support also for a customized LimeSDR firmware and signal source to support PPS samplestamp
     reading.
00041  */
00042 class LimesdrSignalSource : public SignalSourceBase
00043 {
00044 public:
00045     LimesdrSignalSource(const ConfigurationInterface* configuration,
00046         const std::string& role, unsigned int in_stream,
00047         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00048
00049     ~LimesdrSignalSource() = default;
00050
00051     inline size_t item_size() override
00052     {
00053         return item_size_;
```

```
00054     }
00055
00056     void connect(gr::top_block_sptr top_block) override;
00057     void disconnect(gr::top_block_sptr top_block) override;
00058     gr::basic_block_sptr get_left_block() override;
00059     gr::basic_block_sptr get_right_block() override;
00060
00061 private:
00062     gr::limesdr::source::sptr limesdr_source_;
00063     gnss_shared_ptr<gr::block> valve_;
00064     gr::blocks::file_sink::sptr file_sink_;
00065
00066     std::string item_type_;
00067     std::string dump_filename_;
00068     std::string limesdr_serial_;
00069     std::string limesdr_file_;
00070
00071     // Front-end settings
00072     double sample_rate_;
00073     double freq_;
00074     double gain_;
00075     double analog_bw_hz_;
00076     double digital_bw_hz_;
00077     double ext_clock_MHz_;
00078     size_t item_size_;
00079     int64_t samples_;
00080
00081     unsigned int in_stream_;
00082     unsigned int out_stream_;
00083
00084     int limechannel_mode_;
00085     int antenna_;
00086     int channel_;
00087
00088     bool PPS_mode_;
00089     bool dump_;
00090 };
00091
00092
00093 /** \} */
00094 /** \} */
00095 #endif  // GNSS_SDR_LIMESDR_SIGNAL_SOURCE_H
```

13.336 max2771_evkit_signal_source_fpga.h File Reference

Signal source for the MAX2771EVKIT evaluation board connected directly to FPGA accelerators. This source implements only the MAX2771 control. It is NOT compatible with conventional SDR acquisition and tracking blocks.

```
#include "command_event.h"
#include "concurrent_queue.h"
#include "fpga_buffer_monitor.h"
#include "fpga_spidev.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <mutex>
#include <string>
#include <thread>
#include <vector>
```

Classes

- class MAX2771EVKITSignalSourceFPGA

13.336.1 Detailed Description

Signal source for the MAX2771EVKIT evaluation board connected directly to FPGA accelerators. This source implements only the MAX2771 control. It is NOT compatible with conventional SDR acquisition and tracking blocks.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file max2771_evkit_signal_source_fpga.h.

13.337 max2771_evkit_signal_source_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file max2771_evkit_signal_source_fpga.h
00003  * \brief Signal source for the MAX2771EVKIT evaluation board connected directly
00004  * to FPGA accelerators.
00005  * This source implements only the MAX2771 control. It is NOT compatible with
00006  * conventional SDR acquisition and tracking blocks.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_MAX2771_EVKIT_SIGNAL_SOURCE_FPGA_H
00020 #define GNSS_SDR_MAX2771_EVKIT_SIGNAL_SOURCE_FPGA_H
00021
00022 #include "command_event.h"
00023 #include "concurrent_queue.h"
00024 #include "fpga_buffer_monitor.h"
00025 #include "fpga_spidev.h"
00026 #include "gnss_block_interface.h"
00027 #include "signal_source_base.h"
00028 #include <pmt/pmt.h>  // for pmt::pmt_t
00029 #include <cstdint>    // for fixed-width integer types
00030 #include <memory>     // for smart pointers
00031 #include <mutex>      // for mutex
00032 #include <string>     // for strings
00033 #include <thread>     // for threads
00034 #include <vector>     // for std::vector
00035
00036
00037 /** \addtogroup Signal_Source
00038  * \{ */
00039 /** \addtogroup Signal_Source_adapters
00040  * \{ */
00041
00042
00043 class ConfigurationInterface;
00044
00045 class MAX2771EVKITSignalSourceFPGA : public SignalSourceBase
00046 {
00047 public:
00048     MAX2771EVKITSignalSourceFPGA(const ConfigurationInterface *configuration,
00049         const std::string &role, unsigned int in_stream,
00050         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t> *queue);
00051
00052     ~MAX2771EVKITSignalSourceFPGA();
00053
00054     std::vector<uint32_t> setup_regs(uint64_t freq);
00055
00056     inline size_t item_size() override
00057     {
00058         return item_size_;
00059     }
00060
00061     void connect(gr::top_block_sptr top_block) override;
00062     void disconnect(gr::top_block_sptr top_block) override;
00063     gr::basic_block_sptr get_left_block() override;
00064     gr::basic_block_sptr get_right_block() override;
00065
00066 private:
00067     const std::string DEFAULT_BUFF_MON_FILENAME = std::string("FPGA_buffer_monitor_dump.dat");  //
      Default buffer monitor output file
00068     const std::string FREQ_BAND_0_SPI_DEVICE_NAME = std::string("/dev/spidev2.0");              // SPI
      0 device name
00069     const std::string FREQ_BAND_1_SPI_DEVICE_NAME = std::string("/dev/spidev1.0");              // SPI
      1 device name
00070     const uint32_t MAX_NUM_FREQ_BANDS = 2;                                                     //
      Maximum number of frequency bands
00071     const uint32_t DEFAULT_NUM_FREQ_BANDS = 1;                                                 //
      Default number of frequency bands used
00072     const uint64_t DEFAULT_BANDWIDTH = 16400000;                                               //
```

```
    Default bandwidth
00073     const uint32_t DEFAULT_FILTER_ORDER = 5;                                                    //
    Default filter order
00074     const uint64_t DEFAULT_SAMPLING_RATE = 16368000;                                            //
    Default sampling rate
00075     const uint32_t DEFAULT_PGA_GAIN_VALUE = 0x3A;                                               //
    default PGA gain when AGC is off
00076     // max PGA gain value
00077     const uint32_t MAX_PGA_GAIN_VALUE = 0x3F;
00078     // check buffer overflow and perform buffer monitoring every 1s by default
00079     const uint32_t BUFFER_MONITOR_PERIOD_MS = 1000;
00080     // buffer overflow and buffer monitoring initial delay
00081     const uint32_t BUFFER_MONITOR_INITIAL_DELAY_MS = 2000;
00082     // MAX2771 number of configuration registers
00083     const uint32_t MAX2771_NUM_REGS = 11;
00084     // MAX2771 configuration register fields
00085     const uint32_t IDLE = 0x0;                  // Idle mode disabled
00086     const uint32_t MIXPOLE = 0x0;               // set the passive filter pole at mixer output at 13
    MHz.
00087     const uint32_t MIXERMODE_HIGH_BAND = 0x0;  // L1 band enabled
00088     const uint32_t MIXERMODE_LOW_BAND = 0X1;   // L2/L5 band enabled
00089     const uint32_t FCEN = 0x58;                 // Center frequency not used when in low-pass filter
    mode. Set to default value.
00090     const uint32_t FCENX = 0x0;                 // POlyphase filter selection set to Lowpass filter
00091     const uint32_t ANAIMON = 0x0;               // analog monitor disabled
00092     const uint32_t IQEN = 0x1;                  // I and Q channels enable
00093     const uint32_t GAINREF = 0xAA;              // AGC Gain ref
00094     const uint32_t SPI_SDIO_CONFIG = 0x0;       // SPI SDIO config when tri-stated: nothing applied
00095     const uint32_t FORMAT = 0x1;                // sign and magnitude
00096     const uint32_t BITS = 0x2;                  // number of bits in the ADC = 2
00097     const uint32_t DRVCFG = 0x0;                // output driver configuration = CMOS Logic
00098     const uint32_t DIEID = 0x0;                 // identifies version of IC
00099     const uint32_t HILOADEN = 0x0;              // disable output driver for high loads
00100     const uint32_t FHIPEN = 0x1;                // enable highpass coupling between filter and PGA.
00101     const uint32_t PGAIEN = 0x1;                // I-Channel PGA Enable
00102     const uint32_t PGAQEN = 0x1;                // Q-Channel PGA Enable
00103     const uint32_t STRMEN = 0x0;                // disable DSP interface for serial streaming of data
00104     const uint32_t STRMSTART = 0x0;             // the rising edge of this bit enables data streaming
    to the output, clock, data, sync and frame sync outputs.
00105     const uint32_t STRMSTOP = 0x0;              // the rising edge of this bit disables data streaming
    to the output, clock,  data sync and frame sync outputs.
00106     const uint32_t STRMBITS = 0x1;              // number of bits to be streamed: I MSB, I LSB
00107     const uint32_t STAMPEN = 0x1;               // enable frame number insertion
00108     const uint32_t TIMESYNCEN = 0x1;            // enable the output of the time sync pulses at all
    times when streaming is enabled.
00109     const uint32_t DATASYNCEN = 0x0;            // disable the sync pulses at the DATASYNC output
00110     const uint32_t STRMRST = 0x0;               // counter reset not active
00111     const uint32_t LOBAND_L1 = 0x0;             // L1 band
00112     const uint32_t LOBAND_L5 = 0x1;             // L5 band
00113     const uint32_t REFOUTEN = 0x1;              // Output clock buffer enable
00114     const uint32_t IXTAL = 0x1;                 // XTAL osscillator/buffer set to normal current
00115     const uint32_t ICP = 0x0;                   // charge pump current selection set to 0.5 mA
00116     const uint32_t INT_PLL = 0x1;               // PLL mode set to integer-N PLL
00117     const uint32_t PWRSAV = 0x0;                // PLL power save mode disabled
00118     const uint32_t RDIV = 0x10;                 // Set the PLL reference division ratio
00119     const uint32_t FDIV = 0x80000;              // PLL fractional division ratio not used. Set to
    default value
00120     const uint32_t EXTADCCLK = 0x0;             // use internally generated clock
00121     const uint32_t REFCLK_L_CNT = 0x100;        // set the L counter of the reference clock
    configuration to its default value
00122     const uint32_t REFCLK_M_CNT = 0x61B;        // set the M counter of the reference clock
    configuration to its default value
00123     const uint32_t FCLKIN = 0x0;                // fractional clock divider set to default value
00124     const uint32_t ADCCLK = 0x0;                // ADC clock selection set to reference clock
    divider/multiplier
00125     const uint32_t MODE = 0x0;                  // DSP interface mode selection
00126     const uint32_t ADCCLK_L_CNT = 0x100;        // set the L counter of the ADC clock configuration to
    its default value
00127     const uint32_t ADCCLK_M_CNT = 0x61B;        // set the M counter of the ADC clock configuration to
    its default value
00128     const uint32_t PRE_FRACDIV_SEL = 0x0;       // bypass fractional clock divider
00129     const uint32_t CLKOUT_SEL = 0x1;            // CLKOUT selection set to ADC clock
00130     // MAX2771 configuration register registers
00131     const uint32_t TEST_MODE_1_REG_VAL = 0x01E0F401;  // reserved
00132     const uint32_t TEST_MODE_2_REG_VAL = 0x00000002;
00133
00134     bool configure(std::vector<uint32_t> register_values);
00135     void run_buffer_monitor_process();
00136
00137     mutable std::mutex buffer_monitor_mutex;
00138
00139     std::thread thread_buffer_monitor;
00140
00141     std::shared_ptr<Fpga_buffer_monitor> buffer_monitor_fpga;
00142     std::shared_ptr<Fpga_spidev> spidev_fpga;
00143
00144     uint64_t freq_;   // Tuning frequency in single-band mode
```

```
00145     uint64_t freq0_;  // Tuning frequency for band 0 when dual-band mode is enabled
00146     uint64_t freq1_;  // Tuning frequency for band 1 when dual-band mode is enabled
00147     uint64_t sample_rate_;
00148
00149     uint32_t RF_channels_;
00150     uint32_t in_stream_;
00151     uint32_t out_stream_;
00152     uint32_t bandwidth_;      // 2500000, 4200000, 8700000, 16400000, 23400000, 36000000
00153     uint32_t filter_order_;   // 3, 5
00154     uint32_t gain_in_;        // 0 to 0x3F
00155
00156     size_t item_size_;  // 1
00157
00158     bool chipen_;           // chip enable
00159     bool if_filter_gain_;   // true, false
00160     bool LNA_active_;       // true, false
00161     bool enable_agc_;       // true, false
00162     bool enable_ovf_check_buffer_monitor_active_;
00163     bool dump_;
00164     bool rf_shutdown_;
00165 };
00166
00167
00168 /** \} */
00169 /** \} */
00170 #endif  // GNSS_SDR_MAX2771_EVKIT_SIGNAL_SOURCE_FPGA_H
```

13.338 multichannel_file_signal_source.h File Reference

Implementation of a class that reads signals samples from files at different frequency band and adapts it to a SignalSourceInterface.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/file_source.h>
#include <gnuradio/blocks/throttle.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class MultichannelFileSignalSource

  Class that reads signals samples from files at different frequency bands and adapts it to a SignalSourceInterface.

13.338.1 Detailed Description

Implementation of a class that reads signals samples from files at different frequency band and adapts it to a SignalSourceInterface.

Author

Javier Arribas, 2019 jarribas(at)cttc.es

This class represents a file signal source. Internally it uses a GNU Radio's gr_file_source as a connector to the data.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file multichannel_file_signal_source.h.

13.339 multichannel_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file multichannel_file_signal_source.h
00003  * \brief Implementation of a class that reads signals samples from files at
00004  * different frequency band and adapts it to a SignalSourceInterface
00005  * \author Javier Arribas, 2019 jarribas(at)cttc.es
00006  *
00007  * This class represents a file signal source. Internally it uses a GNU Radio's
00008  * gr_file_source as a connector to the data.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_MULTICHANNEL_FILE_SIGNAL_SOURCE_H
00022 #define GNSS_SDR_MULTICHANNEL_FILE_SIGNAL_SOURCE_H
00023
00024 #include "concurrent_queue.h"
00025 #include "gnss_block_interface.h"
00026 #include "signal_source_base.h"
00027 #include <gnuradio/blocks/file_sink.h>
00028 #include <gnuradio/blocks/file_source.h>
00029 #include <gnuradio/blocks/throttle.h>
00030 #include <gnuradio/hier_block2.h>
00031 #include <pmt/pmt.h>
00032 #include <cstdint>
00033 #include <string>
00034 #include <vector>
00035
00036
00037 /** \addtogroup Signal_Source
00038  * \{ */
00039 /** \addtogroup Signal_Source_adapters
00040  * \{ */
00041
00042
00043 class ConfigurationInterface;
00044
00045 /*!
00046  * \brief Class that reads signals samples from files at different frequency bands
00047  * and adapts it to a SignalSourceInterface
00048  */
00049 class MultichannelFileSignalSource : public SignalSourceBase
00050 {
00051 public:
00052     MultichannelFileSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00053         unsigned int in_streams, unsigned int out_streams,
00054         Concurrent_Queue<pmt::pmt_t>* queue);
00055
00056     ~MultichannelFileSignalSource() = default;
00057
00058     inline size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block() override;
00067
00068     inline std::string filename() const
00069     {
00070         return filename_vec_.at(0);
00071     }
00072
00073     inline std::string item_type() const
00074     {
00075         return item_type_;
00076     }
00077
00078     inline bool repeat() const
00079     {
00080         return repeat_;
00081     }
00082
00083     inline int64_t sampling_frequency() const
00084     {
00085         return sampling_frequency_;
00086     }
00087
```

```
00088     inline uint64_t samples() const
00089     {
00090         return samples_;
00091     }
00092
00093 private:
00094     std::vector<gr::blocks::file_source::sptr> file_source_vec_;
00095     gnss_shared_ptr<gr::block> valve_;
00096     gr::blocks::file_sink::sptr sink_;
00097     std::vector<gr::blocks::throttle::sptr> throttle_vec_;
00098     std::vector<std::string> filename_vec_;
00099     std::string item_type_;
00100     uint64_t samples_;
00101     int64_t sampling_frequency_;
00102     size_t item_size_;
00103     int32_t n_channels_;
00104     uint32_t in_streams_;
00105     uint32_t out_streams_;
00106     bool repeat_;
00107     // Throttle control
00108     bool enable_throttle_control_;
00109 };
00110
00111
00112 /** \} */
00113 /** \} */
00114 #endif  // GNSS_SDR_MULTICHANNEL_FILE_SIGNAL_SOURCE_H
```

13.340 nsr_file_signal_source.h File Reference

Implementation of a class that reads signals samples from a NSR 2 bits sampler front-end file and adapts it to a SignalSourceInterface. More information about the front-end here http://www.ifen.com/products/↩ sx-scientific-gnss-solutions/nsr-software-receiver.html.

```
#include "file_source_base.h"
#include "unpack_byte_2bit_samples.h"
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class NsrFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.340.1 Detailed Description

Implementation of a class that reads signals samples from a NSR 2 bits sampler front-end file and adapts it to a SignalSourceInterface. More information about the front-end here http://www.ifen.com/products/↩ sx-scientific-gnss-solutions/nsr-software-receiver.html.

Author

Javier Arribas, 2013 jarribas(at)cttc.es

This class represents a file signal source.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file nsr_file_signal_source.h.

13.341 nsr_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nsr_file_signal_source.h
00003  * \brief Implementation of a class that reads signals samples from a NSR 2 bits sampler front-end
      file
00004  * and adapts it to a SignalSourceInterface. More information about the front-end here
00005  * http://www.ifen.com/products/sx-scientific-gnss-solutions/nsr-software-receiver.html
```

```
00006  * \author Javier Arribas, 2013 jarribas(at)cttc.es
00007  *
00008  * This class represents a file signal source.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_NSR_FILE_SIGNAL_SOURCE_H
00022 #define GNSS_SDR_NSR_FILE_SIGNAL_SOURCE_H
00023
00024 #include "file_source_base.h"
00025 #include "unpack_byte_2bit_samples.h"
00026 #include <cstddef>
00027 #include <string>
00028 #include <tuple>
00029
00030 /** \addtogroup Signal_Source
00031  * \{ */
00032 /** \addtogroup Signal_Source_adapters
00033  * \{ */
00034
00035 class ConfigurationInterface;
00036
00037 /*!
00038  * \brief Class that reads signals samples from a file
00039  * and adapts it to a SignalSourceInterface
00040  */
00041 class NsrFileSignalSource : public FileSourceBase
00042 {
00043 public:
00044     NsrFileSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00045         unsigned int in_streams, unsigned int out_streams,
00046         Concurrent_Queue<pmt::pmt_t>* queue);
00047
00048     ~NsrFileSignalSource() = default;
00049
00050 protected:
00051     std::tuple<size_t, bool> itemTypeToSize() override;
00052     double packetsPerSample() const override;
00053     gnss_shared_ptr<gr::block> source() const override;
00054     void create_file_source_hook() override;
00055     void pre_connect_hook(gr::top_block_sptr top_block) override;
00056     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00057
00058 private:
00059     unpack_byte_2bit_samples_sptr unpack_byte_;
00060 };
00061
00062
00063 /** \} */
00064 /** \} */
00065 #endif  // GNSS_SDR_NSR_FILE_SIGNAL_SOURCE_H
```

13.342 ntlab_file_signal_source.h File Reference

Interface of a class that reads signal samples from a file. Each sample is two bits from multiple channels.

```
#include "file_source_base.h"
#include "unpack_ntlab_2bit_samples.h"
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class NTLabFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.342.1 Detailed Description

Interface of a class that reads signal samples from a file. Each sample is two bits from multiple channels.

Author

Pedro Pereira, 2025 pereirapedrocp (at) gmail.com

This class represents a file signal source.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ntlab_file_signal_source.h.

13.343 ntlab_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ntlab_file_signal_source.h
00003  * \brief Interface of a class that reads signal samples from a file. Each
00004  * sample is two bits from multiple channels.
00005  *
00006  * \author Pedro Pereira, 2025 pereirapedrocp (at) gmail.com
00007  *
00008  * This class represents a file signal source.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_NTLAB_FILE_SIGNAL_SOURCE_H
00022 #define GNSS_SDR_NTLAB_FILE_SIGNAL_SOURCE_H
00023
00024 #include "file_source_base.h"
00025 #include "unpack_ntlab_2bit_samples.h"
00026 #include <cstddef>
00027 #include <string>
00028 #include <tuple>
00029
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 /*!
00040  * \brief Class that reads signals samples from a file
00041  * and adapts it to a SignalSourceInterface
00042  */
00043 class NTLabFileSignalSource : public FileSourceBase
00044 {
00045 public:
00046     NTLabFileSignalSource(const ConfigurationInterface* configuration,
00047         const std::string& role, unsigned int in_streams,
00048         unsigned int out_streams, Concurrent_Queue<pmt::pmt_t>* queue);
00049
00050     ~NTLabFileSignalSource() = default;
00051
00052     gr::basic_block_sptr get_left_block() override;
00053     gr::basic_block_sptr get_right_block() override;
00054
00055 protected:
00056     std::tuple<size_t, bool> itemTypeToSize() override;
00057     double packetsPerSample() const override;
00058     gnss_shared_ptr<gr::block> source() const override;
00059     void create_file_source_hook() override;
00060     void pre_connect_hook(gr::top_block_sptr top_block) override;
00061     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00062
00063 private:
00064     std::string sample_type_;
00065     unpack_ntlab_2bit_samples_sptr unpack_samples_;
```

```
00066     int n_channels_;
00067 };
00068
00069
00070 /** \} */
00071 /** \} */
00072 #endif  // GNSS_SDR_NTLAB_FILE_SIGNAL_SOURCE_H
```

13.344 osmosdr_signal_source.h File Reference

Signal source wrapper for OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki for more information).

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <osmosdr/source.h>
#include <stdexcept>
#include <string>
```

Classes

- class OsmosdrSignalSource

  This class reads samples OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki).

13.344.1 Detailed Description

Signal source wrapper for OsmoSDR-compatible front-ends, such as HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers (see https://osmocom.org/projects/rtl-sdr/wiki for more information).

Author

Javier Arribas, 2012. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file osmosdr_signal_source.h.

13.345 osmosdr_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osmosdr_signal_source.h
00003  * \brief Signal source wrapper for OsmoSDR-compatible front-ends, such as
00004  * HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers
00005  * (see https://osmocom.org/projects/rtl-sdr/wiki for more information)
00006  * \author Javier Arribas, 2012. jarribas(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_OSMOSDR_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_OSMOSDR_SIGNAL_SOURCE_H
00021
00022 #include "concurrent_queue.h"
00023 #include "signal_source_base.h"
```

```
00024 #include <gnuradio/blocks/file_sink.h>
00025 #include <pmt/pmt.h>
00026 #include <cstdint>
00027 #include <memory>
00028 #include <osmosdr/source.h>
00029 #include <stdexcept>
00030 #include <string>
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_adapters
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class reads samples OsmoSDR-compatible front-ends, such as
00042  * HackRF or Realtek's RTL2832U-based USB dongle DVB-T receivers
00043  * (see https://osmocom.org/projects/rtl-sdr/wiki)
00044  */
00045 class OsmosdrSignalSource : public SignalSourceBase
00046 {
00047 public:
00048     OsmosdrSignalSource(const ConfigurationInterface* configuration,
00049         const std::string& role, unsigned int in_stream,
00050         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00051
00052     ~OsmosdrSignalSource() = default;
00053
00054     inline size_t item_size() override
00055     {
00056         return item_size_;
00057     }
00058
00059     void connect(gr::top_block_sptr top_block) override;
00060     void disconnect(gr::top_block_sptr top_block) override;
00061     gr::basic_block_sptr get_left_block() override;
00062     gr::basic_block_sptr get_right_block() override;
00063
00064 private:
00065     void driver_instance();
00066
00067     osmosdr::source::sptr osmosdr_source_;
00068     gnss_shared_ptr<gr::block> valve_;
00069     gr::blocks::file_sink::sptr file_sink_;
00070
00071     std::string item_type_;
00072     std::string dump_filename_;
00073     std::string osmosdr_args_;
00074     std::string antenna_;
00075
00076     // Front-end settings
00077     double sample_rate_;
00078     double freq_;
00079     double gain_;
00080     double if_gain_;
00081     double rf_gain_;
00082     double if_bw_;
00083
00084     size_t item_size_;
00085     int64_t samples_;
00086
00087     unsigned int in_stream_;
00088     unsigned int out_stream_;
00089
00090     bool AGC_enabled_;
00091     bool dump_;
00092 };
00093
00094
00095 /** \} */
00096 /** \} */
00097 #endif  // GNSS_SDR_OSMOSDR_SIGNAL_SOURCE_H
```

13.346 plutosdr_signal_source.h File Reference

Signal source for PlutoSDR.

```
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <iio/pluto_source.h>
#include "concurrent_queue.h"
```

```
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
```

Classes

- class PlutosdrSignalSource

13.346.1 Detailed Description

Signal source for PlutoSDR.

Author

Rodrigo Muñoz, 2017, rmunozl(at)inacap.cl, rodrigo.munoz(at)proteinlab.cl

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file plutosdr_signal_source.h.

13.347 plutosdr_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file plutosdr_signal_source.h
00003  * \brief Signal source for PlutoSDR
00004  * \author Rodrigo Muñoz, 2017, rmunozl(at)inacap.cl, rodrigo.munoz(at)proteinlab.cl
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_PLUTOSDR_SIGNAL_SOURCE_H
00020 #define GNSS_SDR_PLUTOSDR_SIGNAL_SOURCE_H
00021
00022 #include "signal_source_base.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #if GRIIO_INCLUDE_HAS_GNURADIO
00025 #if GR_IIO_TEMPLATIZED_API
00026 #include <gnuradio/iio/fmcomms2_source.h>
00027 #else
00028 #include <gnuradio/iio/pluto_source.h>
00029 #endif
00030 #else
00031 #include <iio/pluto_source.h>
00032 #endif
00033 #include "concurrent_queue.h"
00034 #include <pmt/pmt.h>
00035 #include <cstdint>
00036 #include <string>
00037
00038
00039 /** \addtogroup Signal_Source
00040  * \{ */
00041 /** \addtogroup Signal_Source_adapters
00042  * \{ */
00043
00044
00045 class ConfigurationInterface;
00046
00047 /*!
00048  */
00049 class PlutosdrSignalSource : public SignalSourceBase
00050 {
00051 public:
00052     PlutosdrSignalSource(const ConfigurationInterface* configuration,
00053         const std::string& role, unsigned int in_stream,
```

```
00054         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00055
00056     ~PlutosdrSignalSource() = default;
00057
00058     size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block() override;
00067
00068 private:
00069     const std::string default_gain_mode = std::string("slow_attack");
00070 #if GR_IIO_TEMPLATIZED_API
00071     gr::iio::fmcomms2_source<gr_complex>::sptr plutosdr_source_;
00072 #else
00073     gr::iio::pluto_source::sptr plutosdr_source_;
00074 #endif
00075
00076     gnss_shared_ptr<gr::block> valve_;
00077     gr::blocks::file_sink::sptr file_sink_;
00078
00079     std::string dump_filename_;
00080
00081     // Front-end settings
00082     std::string uri_;  // device direction
00083     std::string gain_mode_;
00084     std::string filter_file_;
00085     std::string filter_source_;
00086     std::string filter_filename_;
00087     std::string item_type_;
00088     double rf_gain_;
00089     int64_t samples_;
00090     uint64_t freq_;  // frequency of local oscillator
00091     uint64_t sample_rate_;
00092     uint64_t bandwidth_;
00093     uint64_t buffer_size_;  // reception buffer
00094     size_t item_size_;
00095     float Fpass_;
00096     float Fstop_;
00097     unsigned int in_stream_;
00098     unsigned int out_stream_;
00099
00100     bool quadrature_;
00101     bool rf_dc_;
00102     bool bb_dc_;
00103     bool filter_auto_;
00104     bool dump_;
00105 };
00106
00107
00108 /** \} */
00109 /** \} */
00110 #endif  // GNSS_SDR_PLUTOSDR_SIGNAL_SOURCE_H
```

13.348 pocket_sdr_signal_source.h File Reference

Signal source for Pocket SDR front-ends.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/pocketsdr/pocketsdr_source.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <string>
```

Classes

- class PocketSdrSignalSource

  This class instantiates the gr-pocketsdr signal source for the Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends.

13.348.1 Detailed Description

Signal source for Pocket SDR front-ends.

Author

Minhaj Ahmad, 2026. mahmad12(at)crimson.ua.edu

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pocket_sdr_signal_source.h.

13.349 pocket_sdr_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pocket_sdr_signal_source.h
00003  * \brief Signal source for Pocket SDR front-ends
00004  * \author Minhaj Ahmad, 2026. mahmad12(at)crimson.ua.edu
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_POCKET_SDR_SIGNAL_SOURCE_H
00018 #define GNSS_SDR_POCKET_SDR_SIGNAL_SOURCE_H
00019
00020 #include "concurrent_queue.h"
00021 #include "gnss_block_interface.h"
00022 #include "signal_source_base.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <gnuradio/pocketsdr/pocketsdr_source.h>
00025 #include <pmt/pmt.h>
00026 #include <cstdint>
00027 #include <memory>
00028 #include <string>
00029
00030 /** \addtogroup Signal_Source
00031  * \{ */
00032 /** \addtogroup Signal_Source_adapters
00033  * \{ */
00034
00035
00036 class ConfigurationInterface;
00037
00038 /*!
00039  * \brief This class instantiates the gr-pocketsdr signal source for the
00040  * Pocket SDR FE 2CH/4CH/8CH GNSS RF front-ends.
00041  */
00042 class PocketSdrSignalSource : public SignalSourceBase
00043 {
00044 public:
00045     PocketSdrSignalSource(const ConfigurationInterface* configuration,
00046         const std::string& role, unsigned int in_stream,
00047         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00048
00049     ~PocketSdrSignalSource() = default;
00050
00051     inline size_t item_size() override
00052     {
00053         return item_size_;
00054     }
00055
00056     void connect(gr::top_block_sptr top_block) override;
00057     void disconnect(gr::top_block_sptr top_block) override;
00058     gr::basic_block_sptr get_left_block() override;
00059     gr::basic_block_sptr get_right_block() override;
00060
00061 private:
00062     gr::pocketsdr::pocketsdr_source::sptr pocketsdr_source_;
00063     gnss_shared_ptr<gr::block> valve_;
00064     gr::blocks::file_sink::sptr file_sink_;
00065
00066     std::string item_type_;
00067     std::string dump_filename_;
```

```
00068     std::string conf_file_;
00069
00070     double sampling_frequency_;
00071     double freq_;
00072     size_t item_size_;
00073     int64_t samples_;
00074
00075     unsigned int in_stream_;
00076     unsigned int out_stream_;
00077
00078     int channel_;
00079
00080     bool dump_;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_POCKET_SDR_SIGNAL_SOURCE_H
```

13.350 raw_array_signal_source.h File Reference

CTTC Experimental GNSS 8 channels array signal source.

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/hier_block2.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <memory>
#include <string>
```

Classes

- class RawArraySignalSource

  This class reads samples from an antenna array RF front-end signal sampler.

13.350.1 Detailed Description

CTTC Experimental GNSS 8 channels array signal source.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file raw_array_signal_source.h.

13.351 raw_array_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file raw_array_signal_source.h
00003  * \brief CTTC Experimental GNSS 8 channels array signal source
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
```

```
00018 #ifndef GNSS_SDR_RAW_ARRAY_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_RAW_ARRAY_SIGNAL_SOURCE_H
00020
00021 #include "concurrent_queue.h"
00022 #include "signal_source_base.h"
00023 #include <gnuradio/blocks/file_sink.h>
00024 #include <gnuradio/hier_block2.h>
00025 #include <pmt/pmt.h>
00026 #include <cstdint>
00027 #include <memory>
00028 #include <string>
00029
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 /*!
00040  * \brief This class reads samples from an antenna array RF front-end signal sampler
00041  */
00042 class RawArraySignalSource : public SignalSourceBase
00043 {
00044 public:
00045     RawArraySignalSource(const ConfigurationInterface* configuration,
00046         std::string role, unsigned int in_stream,
00047         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00048
00049     ~RawArraySignalSource() = default;
00050
00051     inline size_t item_size() override
00052     {
00053         return item_size_;
00054     }
00055
00056     void connect(gr::top_block_sptr top_block) override;
00057     void disconnect(gr::top_block_sptr top_block) override;
00058     gr::basic_block_sptr get_left_block() override;
00059     gr::basic_block_sptr get_right_block() override;
00060
00061 private:
00062     gr::block_sptr raw_array_source_;
00063     gr::blocks::file_sink::sptr file_sink_;
00064     std::string item_type_;
00065     std::string dump_filename_;
00066     std::string eth_device_;
00067     size_t item_size_;
00068     [[maybe_unused]] int64_t samples_;
00069     unsigned int in_stream_;
00070     unsigned int out_stream_;
00071     bool dump_;
00072 };
00073
00074
00075 /** \} */
00076 /** \} */
00077 #endif  // GNSS_SDR_RAW_ARRAY_SIGNAL_SOURCE_H
```

13.352 rtl_tcp_signal_source.h File Reference

Signal source which reads from rtl_tcp. (see https://osmocom.org/projects/rtl-sdr/wiki for more information).

```
#include "concurrent_queue.h"
#include "rtl_tcp_signal_source_c.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/deinterleave.h>
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/float_to_complex.h>
#include <pmt/pmt.h>
#include <stdexcept>
#include <string>
```

Classes

- class RtlTcpSignalSource

  This class reads from rtl_tcp, which streams interleaved I/Q samples over TCP. (see https://osmocom.org/projects/rtl-sdr/wiki).

13.352.1 Detailed Description

Signal source which reads from rtl_tcp. (see https://osmocom.org/projects/rtl-sdr/wiki for more information).

Author

Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file rtl_tcp_signal_source.h.

13.353 rtl_tcp_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtl_tcp_signal_source.h
00003  * \brief Signal source which reads from rtl_tcp.
00004  * (see https://osmocom.org/projects/rtl-sdr/wiki for more information)
00005  * \author Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_H
00020
00021 #include "concurrent_queue.h"
00022 #include "rtl_tcp_signal_source_c.h"
00023 #include "signal_source_base.h"
00024 #include <gnuradio/blocks/deinterleave.h>
00025 #include <gnuradio/blocks/file_sink.h>
00026 #include <gnuradio/blocks/float_to_complex.h>
00027 #include <pmt/pmt.h>
00028 #include <stdexcept>
00029 #include <string>
00030
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_adapters
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class reads from rtl_tcp, which streams interleaved
00042  * I/Q samples over TCP.
00043  * (see https://osmocom.org/projects/rtl-sdr/wiki)
00044  */
00045 class RtlTcpSignalSource : public SignalSourceBase
00046 {
00047 public:
00048     RtlTcpSignalSource(const ConfigurationInterface* configuration,
00049         const std::string& role,
00050         unsigned int in_stream,
00051         unsigned int out_stream,
00052         Concurrent_Queue<pmt::pmt_t>* queue);
00053
00054     ~RtlTcpSignalSource() = default;
00055
00056     inline size_t item_size() override
```

```
00057     {
00058         return item_size_;
00059     }
00060
00061     void connect(gr::top_block_sptr top_block) override;
00062     void disconnect(gr::top_block_sptr top_block) override;
00063     gr::basic_block_sptr get_left_block() override;
00064     gr::basic_block_sptr get_right_block() override;
00065
00066 private:
00067     void MakeBlock();
00068
00069     rtl_tcp_signal_source_c_sptr signal_source_;
00070
00071     gnss_shared_ptr<gr::block> valve_;
00072     gr::blocks::file_sink::sptr file_sink_;
00073
00074     std::string item_type_;
00075     std::string dump_filename_;
00076
00077     // rtl_tcp settings
00078     std::string address_;
00079     size_t item_size_;
00080     uint64_t samples_;
00081     double rf_gain_;
00082     int sample_rate_;
00083     int freq_;
00084     int gain_;
00085     int if_gain_;
00086     unsigned int in_stream_;
00087     unsigned int out_stream_;
00088     int16_t port_;
00089     bool AGC_enabled_;
00090     bool flip_iq_;
00091     bool dump_;
00092 };
00093
00094
00095 /** \} */
00096 /** \} */
00097 #endif  // GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_H
```

13.354 signal_source_base.h File Reference

Header file of the base class to signal_source GNSS blocks.

```
#include "signal_source_interface.h"
#include <cstddef>
#include <string>
```

Classes

- class SignalSourceBase

13.354.1 Detailed Description

Header file of the base class to signal_source GNSS blocks.

Author

Jim Melton, 2020. jim.melton(at)sncorp.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file signal_source_base.h.

13.355 signal_source_base.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_source_base.h
00003  * \brief Header file of the base class to signal_source GNSS blocks.
00004  * \author Jim Melton, 2020. jim.melton(at)sncorp.com
```

```
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SIGNAL_SOURCE_BASE_H
00019 #define GNSS_SDR_SIGNAL_SOURCE_BASE_H
00020
00021 #include "signal_source_interface.h"
00022 #include <cstddef>
00023 #include <string>
00024
00025
00026 class ConfigurationInterface;
00027
00028 class SignalSourceBase : public SignalSourceInterface
00029 {
00030 public:
00031     std::string role() final;
00032     std::string implementation() final;
00033
00034     size_t getRfChannels() const override;
00035     gr::basic_block_sptr get_left_block() override;  // non-sensical; implement once
00036
00037 protected:
00038     //! Constructor
00039     SignalSourceBase(ConfigurationInterface const* configuration, std::string role, std::string impl);
00040
00041     //! utility for decoding passed ".item_type" values
00042     //!  @param[in]  item_type - user provided string, should be one of the known types
00043     //!  @param[out] is_interleaved - if non-null, the pointed to memory is updated with
00044     //!                               whether the data is interleaved I/Q (e.g., ishort)
00045     //!  @param[in] throw_on_error  - if true, throw an exception if the string does not
00046     //!                               represent a known type
00047     //!  @return the size in bytes of the passed type
00048     size_t decode_item_type(std::string const& item_type, bool* is_interleaved = nullptr, bool
      throw_on_error = false);
00049
00050 private:
00051     std::string const role_;
00052     std::string const implementation_;
00053     size_t rfChannels_;
00054 };
00055
00056
00057 #endif
```

13.356 spir_file_signal_source.h File Reference

Implementation of a class that reads signals samples from a SPIR file and adapts it to a SignalSourceInterface.

```
#include "file_source_base.h"
#include "unpack_intspir_1bit_samples.h"
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class SpirFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.356.1 Detailed Description

Implementation of a class that reads signals samples from a SPIR file and adapts it to a SignalSourceInterface.

Author

Fran Fabra, 2014 fabra(at)ice.csic.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file spir_file_signal_source.h.

13.357 spir_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file spir_file_signal_source.h
00003  * \brief Implementation of a class that reads signals samples from a SPIR file
00004  * and adapts it to a SignalSourceInterface.
00005  * \author Fran Fabra, 2014 fabra(at)ice.csic.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SPIR_FILE_SIGNAL_SOURCE_H
00019 #define GNSS_SDR_SPIR_FILE_SIGNAL_SOURCE_H
00020
00021 #include "file_source_base.h"
00022 #include "unpack_intspir_1bit_samples.h"
00023 #include <cstddef>
00024 #include <string>
00025 #include <tuple>
00026
00027
00028 /** \addtogroup Signal_Source
00029  * \{ */
00030 /** \addtogroup Signal_Source_adapters
00031  * \{ */
00032
00033
00034 class ConfigurationInterface;
00035
00036 /*!
00037  * \brief Class that reads signals samples from a file
00038  * and adapts it to a SignalSourceInterface
00039  */
00040 class SpirFileSignalSource : public FileSourceBase
00041 {
00042 public:
00043     SpirFileSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00044         unsigned int in_streams, unsigned int out_streams,
00045         Concurrent_Queue<pmt::pmt_t>* queue);
00046
00047     ~SpirFileSignalSource() = default;
00048
00049 protected:
00050     std::tuple<size_t, bool> itemTypeToSize() override;
00051     gnss_shared_ptr<gr::block> source() const override;
00052     void create_file_source_hook() override;
00053     void pre_connect_hook(gr::top_block_sptr top_block) override;
00054     void post_disconnect_hook(gr::top_block_sptr top_block) override;
00055
00056
00057 private:
00058     unpack_intspir_1bit_samples_sptr unpack_intspir_;
00059 };
00060
00061
00062 /** \} */
00063 /** \} */
00064 #endif  // GNSS_SDR_SPIR_FILE_SIGNAL_SOURCE_H
```

13.358 two_bit_cpx_file_signal_source.h File Reference

Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface.

```
#include "file_source_base.h"
#include "unpack_byte_2bit_cpx_samples.h"
#include <gnuradio/blocks/interleaved_short_to_complex.h>
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class TwoBitCpxFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.358.1 Detailed Description

Interface of a class that reads signals samples from a 2 bit complex sampler front-end file and adapts it to a SignalSourceInterface.

Author

Javier Arribas, 2015 jarribas(at)cttc.es

This class represents a file signal source.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file two_bit_cpx_file_signal_source.h.

13.359 two_bit_cpx_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file two_bit_cpx_file_signal_source.h
00003  * \brief Interface of a class that reads signals samples from a 2 bit complex sampler front-end file
00004  * and adapts it to a SignalSourceInterface.
00005  * \author Javier Arribas, 2015 jarribas(at)cttc.es
00006  *
00007  * This class represents a file signal source.
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_TWO_BIT_CPX_FILE_SIGNAL_SOURCE_H
00021 #define GNSS_SDR_TWO_BIT_CPX_FILE_SIGNAL_SOURCE_H
00022
00023 #include "file_source_base.h"
00024 #include "unpack_byte_2bit_cpx_samples.h"
00025 #include <gnuradio/blocks/interleaved_short_to_complex.h>
00026 #include <cstddef>
00027 #include <string>
00028 #include <tuple>
00029
00030 /** \addtogroup Signal_Source
00031  * \{ */
00032 /** \addtogroup Signal_Source_adapters
00033  * \{ */
00034
00035
00036 class ConfigurationInterface;
00037
00038 /*!
00039  * \brief Class that reads signals samples from a file
00040  * and adapts it to a SignalSourceInterface
00041  */
00042 class TwoBitCpxFileSignalSource : public FileSourceBase
00043 {
```

```
00044 public:
00045     TwoBitCpxFileSignalSource(const ConfigurationInterface* configuration,
00046         const std::string& role,
00047         unsigned int in_streams,
00048         unsigned int out_streams,
00049         Concurrent_Queue<pmt::pmt_t>* queue);
00050
00051     ~TwoBitCpxFileSignalSource() = default;
00052
00053 protected:
00054     std::tuple<size_t, bool> itemTypeToSize() override;
00055     double packetsPerSample() const override;
00056     gnss_shared_ptr<gr::block> source() const override;
00057     void create_file_source_hook() override;
00058     void pre_connect_hook(gr::top_block_sptr top_block) override;
00059     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00060
00061 private:
00062     unpack_byte_2bit_cpx_samples_sptr unpack_byte_;
00063     gr::blocks::interleaved_short_to_complex::sptr inter_shorts_to_cpx_;
00064 };
00065
00066
00067 /** \} */
00068 /** \} */
00069 #endif  // GNSS_SDR_TWO_BIT_CPX_FILE_SIGNAL_SOURCE_H
```

13.360 two_bit_packed_file_signal_source.h File Reference

Interface of a class that reads signals samples from a file. Each sample is two bits, which are packed into bytes or shorts.

```
#include "file_source_base.h"
#include "unpack_2bit_samples.h"
#include <gnuradio/blocks/interleaved_char_to_complex.h>
#include <cstddef>
#include <string>
#include <tuple>
```

Classes

- class TwoBitPackedFileSignalSource

  Class that reads signals samples from a file and adapts it to a SignalSourceInterface.

13.360.1 Detailed Description

Interface of a class that reads signals samples from a file. Each sample is two bits, which are packed into bytes or shorts.

Author

Cillian O'Driscoll, 2015 cillian.odriscoll (at) gmail.com

This class represents a file signal source.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file two_bit_packed_file_signal_source.h.

13.361 two_bit_packed_file_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file two_bit_packed_file_signal_source.h
00003  * \brief Interface of a class that reads signals samples from a file. Each
00004  * sample is two bits, which are packed into bytes or shorts.
00005  *
00006  * \author Cillian O'Driscoll, 2015 cillian.odriscoll (at) gmail.com
00007  *
00008  * This class represents a file signal source.
```

```
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_TWO_BIT_PACKED_FILE_SIGNAL_SOURCE_H
00022 #define GNSS_SDR_TWO_BIT_PACKED_FILE_SIGNAL_SOURCE_H
00023
00024 #include "file_source_base.h"
00025 #include "unpack_2bit_samples.h"
00026 #include <gnuradio/blocks/interleaved_char_to_complex.h>
00027 #include <cstddef>
00028 #include <string>
00029 #include <tuple>
00030
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_adapters
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief Class that reads signals samples from a file
00042  * and adapts it to a SignalSourceInterface
00043  */
00044 class TwoBitPackedFileSignalSource : public FileSourceBase
00045 {
00046 public:
00047     TwoBitPackedFileSignalSource(const ConfigurationInterface* configuration, const std::string& role,
00048         unsigned int in_streams, unsigned int out_streams,
00049         Concurrent_Queue<pmt::pmt_t>* queue);
00050
00051     ~TwoBitPackedFileSignalSource() = default;
00052
00053 private:
00054     inline bool big_endian_items() const
00055     {
00056         return big_endian_items_;
00057     }
00058
00059     inline bool big_endian_bytes() const
00060     {
00061         return big_endian_bytes_;
00062     }
00063
00064
00065     inline bool reverse_interleaving() const
00066     {
00067         return reverse_interleaving_;
00068     }
00069
00070 protected:
00071     std::tuple<size_t, bool> itemTypeToSize() override;
00072     double packetsPerSample() const override;
00073     gnss_shared_ptr<gr::block> source() const override;
00074     void create_file_source_hook() override;
00075     void pre_connect_hook(gr::top_block_sptr top_block) override;
00076     void pre_disconnect_hook(gr::top_block_sptr top_block) override;
00077
00078 private:
00079     std::string sample_type_;
00080     bool big_endian_items_;
00081     bool big_endian_bytes_;
00082     bool reverse_interleaving_;
00083     unpack_2bit_samples_sptr unpack_samples_;
00084     gnss_shared_ptr<gr::block> char_to_float_;
00085 };
00086
00087
00088 /** \} */
00089 /** \} */
00090 #endif  // GNSS_SDR_TWO_BIT_CPX_FILE_SIGNAL_SOURCE_H
```

13.362 uhd_signal_source.h File Reference

Interface for the Universal Hardware Driver signal source.

```
#include "concurrent_queue.h"
#include "signal_source_base.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/hier_block2.h>
#include <gnuradio/uhd/usrp_source.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class UhdSignalSource

  This class reads samples from a UHD device (see http://code.ettus.com/redmine/ettus/↩ projects/uhd/wiki).

13.362.1 Detailed Description

Interface for the Universal Hardware Driver signal source.

Author

Javier Arribas, 2012. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file uhd_signal_source.h.

13.363 uhd_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file uhd_signal_source.h
00003  * \brief Interface for the Universal Hardware Driver signal source
00004  * \author Javier Arribas, 2012. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_UHD_SIGNAL_SOURCE_H
00018 #define GNSS_SDR_UHD_SIGNAL_SOURCE_H
00019
00020 #include "concurrent_queue.h"
00021 #include "signal_source_base.h"
00022 #include <gnuradio/blocks/file_sink.h>
00023 #include <gnuradio/hier_block2.h>
00024 #include <gnuradio/uhd/usrp_source.h>
00025 #include <pmt/pmt.h>
00026 #include <cstdint>
00027 #include <string>
00028 #include <vector>
00029
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_adapters
00034  * \{ */
00035
00036 class ConfigurationInterface;
```

```
00037
00038 /*!
00039  * \brief This class reads samples from a UHD device (see
      http://code.ettus.com/redmine/ettus/projects/uhd/wiki)
00040  */
00041 class UhdSignalSource : public SignalSourceBase
00042 {
00043 public:
00044     UhdSignalSource(const ConfigurationInterface* configuration,
00045         const std::string& role, unsigned int in_stream,
00046         unsigned int out_stream, Concurrent_Queue<pmt::pmt_t>* queue);
00047
00048     ~UhdSignalSource() = default;
00049
00050     inline size_t item_size() override
00051     {
00052         return item_size_;
00053     }
00054
00055     void connect(gr::top_block_sptr top_block) override;
00056     void disconnect(gr::top_block_sptr top_block) override;
00057     gr::basic_block_sptr get_left_block() override;
00058     gr::basic_block_sptr get_right_block() override;
00059     gr::basic_block_sptr get_right_block(int RF_channel) override;
00060
00061 private:
00062     gr::uhd::usrp_source::sptr uhd_source_;
00063
00064     std::vector<gnss_shared_ptr<gr::block» valve_;
00065     std::vector<gr::blocks::file_sink::sptr> file_sink_;
00066     std::vector<double> freq_;
00067     std::vector<double> gain_;
00068     std::vector<double> IF_bandwidth_hz_;
00069     std::vector<uint64_t> samples_;
00070     std::vector<std::string> dump_filename_;
00071     std::vector<bool> dump_;
00072
00073     uhd::stream_args_t uhd_stream_args_;  // UHD SETTINGS
00074
00075     std::string device_address_;
00076     std::string item_type_;
00077     std::string subdevice_;
00078     std::string clock_source_;
00079
00080     //    * The OTW format is a string that describes the format over-the-wire.
00081     //    * The following over-the-wire formats have been implemented:
00082     //    * - sc16 - Q16 I16
00083     //    * - sc8 - Q8_1 I8_1 Q8_0 I8_0
00084     //    * - sc12 (Only some devices)
00085     std::string otw_format_;
00086
00087     double sample_rate_;
00088     size_t item_size_;
00089     int RF_channels_;
00090     unsigned int in_stream_;
00091     unsigned int out_stream_;
00092 };
00093
00094
00095 /** \} */
00096 /** \} */
00097 #endif  // GNSS_SDR_UHD_SIGNAL_SOURCE_H
```

13.364 zmq_signal_source.h File Reference

Signal source which reads from ZeroMQ.

```
#include "signal_source_base.h"
#include "concurrent_queue.h"
#include <gnuradio/blocks/file_sink.h>
#include <gnuradio/blocks/vector_to_stream.h>
#include <gnuradio/zeromq/sub_source.h>
#include <pmt/pmt.h>
#include <string>
```

Classes

- class ZmqSignalSource

This class supports the following properties:

13.364.1 Detailed Description

Signal source which reads from ZeroMQ.

Author

Jim Melton, 2022. jim.melton(at)sncorp.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file zmq_signal_source.h.

13.365 zmq_signal_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file zmq_signal_source.h
00003  * \brief Signal source which reads from ZeroMQ.
00004  * \author Jim Melton, 2022. jim.melton(at)sncorp.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_ZMQ_SIGNAL_SOURCE_H
00018 #define GNSS_SDR_ZMQ_SIGNAL_SOURCE_H
00019
00020 #include "signal_source_base.h"
00021 //
00022 #include "concurrent_queue.h"
00023 #include <gnuradio/blocks/file_sink.h>  // for dump
00024 #include <gnuradio/blocks/vector_to_stream.h>
00025 #include <gnuradio/zeromq/sub_source.h>
00026 #include <pmt/pmt.h>
00027 #include <string>
00028
00029 /** \addtogroup Signal_Source
00030  * \{ */
00031 /** \addtogroup Signal_Source_adapters
00032  * \{ */
00033
00034 //! This class supports the following properties:
00035 //!
00036 //!   .endpoint   - the ZMQ endpoint to be connected to
00037 //!   .vlen       - vector length of the input items (default 1, one item)
00038 //!                 this must match the size of the publisher!
00039 //!   .pass_tags  - boolean flag if tags should be propagated (default false)
00040 //!   .timeout_ms - receive timeout, in milliseconds (default 100)
00041 //!   .hwm        - ZMQ high water mark (default -1, ZMQ default)
00042 //!
00043 //!   .item_type - data type of the samples (default "gr_complex")
00044 //!
00045 //! (probably should be abstracted to the base class)
00046 //!
00047 //!   .dump          - whether to archive input data
00048 //!   .dump_filename - if dumping, path to file for output
00049 //!
00050
00051 class ZmqSignalSource : public SignalSourceBase
00052 {
00053 public:
00054     ZmqSignalSource(const ConfigurationInterface* configuration,
00055         const std::string& role,
00056         unsigned int in_stream,
00057         unsigned int out_stream,
00058         Concurrent_Queue<pmt::pmt_t>* queue);
00059
00060     ~ZmqSignalSource() = default;
00061
00062     size_t item_size() override;
00063
```

```
00064     auto connect(gr::top_block_sptr top_block) -> void override;
00065     auto disconnect(gr::top_block_sptr top_block) -> void override;
00066     auto get_right_block() -> gr::basic_block_sptr override;
00067
00068 private:
00069     gr::zeromq::sub_source::sptr d_source_block;
00070     gr::blocks::vector_to_stream::sptr d_vec_block;
00071     gr::blocks::file_sink::sptr d_dump_sink;
00072
00073     size_t d_item_size;
00074     std::string d_dump_filename;
00075     bool d_dump;
00076 };
00077
00078 /** \} */
00079 /** \} */
00080 #endif
```

13.366 ad936x_iio_source.h File Reference

A direct IIO custom front-end gnss-sdr signal gnuradio block for the AD936x AD front-end family with special FPGA custom functionalities.

```
#include "ad936x_iio_custom.h"
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include "ppstcprx.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <iostream>
#include <memory>
#include <string>
#include <thread>
#include <vector>
```

Classes

- class ad936x_iio_source

  This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.

Typedefs

- using ad936x_iio_source_sptr = gnss_shared_ptr<ad936x_iio_source>

Functions

- ad936x_iio_source_sptr **ad936x_iio_make_source_sptr** (const std::string &pluto_uri_, const std::string &board_type_, int64_t bandwidth_, int64_t sample_rate_, int64_t freq_, const std::string &rf_port_select_, const std::string &rf_filter, const std::string &gain_mode_rx0_, const std::string &gain_mode_rx1_, double rf↩ _gain_rx0_, double rf_gain_rx1_, bool enable_ch0, bool enable_ch1, int64_t freq_2ch, bool ppsmode_, bool customsamplesize_, const std::string &fe_ip_, int fe_ctlport_, int ssize_, int bshift_, bool spattern_, double lo_attenuation_db_, bool high_side_lo_, int tx_lo_channel_)

13.366.1 Detailed Description

A direct IIO custom front-end gnss-sdr signal gnuradio block for the AD936x AD front-end family with special FPGA custom functionalities.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ad936x_iio_source.h.

13.367 ad936x_iio_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ad936x_iio_source.h
00003  * \brief A direct IIO custom front-end gnss-sdr signal gnuradio block for the
00004  * AD936x AD front-end family with special FPGA custom functionalities.
00005  * \author Javier Arribas, jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_AD936X_IIO_SOURCE_H
00020 #define GNSS_SDR_AD936X_IIO_SOURCE_H
00021
00022 #include "ad936x_iio_custom.h"
00023 #include "concurrent_queue.h"
00024 #include "gnss_block_interface.h"
00025 #include "ppstcprx.h"
00026 #include <gnuradio/block.h>
00027 #include <pmt/pmt.h>
00028 #include <cstddef>
00029 #include <cstdint>
00030 #include <fstream>
00031 #include <iostream>
00032 #include <memory>
00033 #include <string>
00034 #include <thread>
00035 #include <vector>
00036
00037 /** \addtogroup Signal_Source
00038  * \{ */
00039 /** \addtogroup Signal_Source_gnuradio_blocks
00040  * \{ */
00041
00042
00043 class ad936x_iio_source;
00044
00045 using ad936x_iio_source_sptr = gnss_shared_ptr<ad936x_iio_source>;
00046
00047 ad936x_iio_source_sptr ad936x_iio_make_source_sptr(
00048     const std::string &pluto_uri_,
00049     const std::string &board_type_,
00050     int64_t bandwidth_,
00051     int64_t sample_rate_,
00052     int64_t freq_,
00053     const std::string &rf_port_select_,
00054     const std::string &rf_filter,
00055     const std::string &gain_mode_rx0_,
00056     const std::string &gain_mode_rx1_,
00057     double rf_gain_rx0_,
00058     double rf_gain_rx1_,
00059     bool enable_ch0,
00060     bool enable_ch1,
00061     int64_t freq_2ch,
00062     bool ppsmode_,
00063     bool customsamplesize_,
00064     const std::string &fe_ip_,
00065     int fe_ctlport_,
00066     int ssize_,
00067     int bshift_,
00068     bool spattern_,
00069     double lo_attenuation_db_,
00070     bool high_side_lo_,
00071     int tx_lo_channel_);
00072
```

```
00073 /*!
00074  * \brief This class implements conversion between Labsat 2, 3 and 3 Wideband
00075  * formats to gr_complex
00076  */
00077 class ad936x_iio_source : public gr::block
00078 {
00079 public:
00080     ~ad936x_iio_source();
00081
00082     //! start the sample transmission
00083     bool start();
00084     //! stop the sample transmission
00085     bool stop();
00086
00087     int general_work(int noutput_items,
00088         gr_vector_int &ninput_items,
00089         gr_vector_const_void_star &input_items,
00090         gr_vector_void_star &output_items);
00091
00092 private:
00093     friend ad936x_iio_source_sptr ad936x_iio_make_source_sptr(
00094         const std::string &pluto_uri_,
00095         const std::string &board_type_,
00096         int64_t bandwidth_,
00097         int64_t sample_rate_,
00098         int64_t freq_,
00099         const std::string &rf_port_select_,
00100         const std::string &rf_filter,
00101         const std::string &gain_mode_rx0_,
00102         const std::string &gain_mode_rx1_,
00103         double rf_gain_rx0_,
00104         double rf_gain_rx1_,
00105         bool enable_ch0,
00106         bool enable_ch1,
00107         int64_t freq_2ch,
00108         bool ppsmode_,
00109         bool customsamplesize_,
00110         const std::string &fe_ip_,
00111         int fe_ctlport_,
00112         int ssize_,
00113         int bshift_,
00114         bool spattern_,
00115         double lo_attenuation_db_,
00116         bool high_side_lo_,
00117         int tx_lo_channel_);
00118
00119     ad936x_iio_source(
00120         const std::string &pluto_uri_,
00121         const std::string &board_type_,
00122         int64_t bandwidth_,
00123         int64_t sample_rate_,
00124         int64_t freq_,
00125         const std::string &rf_port_select_,
00126         const std::string &rf_filter,
00127         const std::string &gain_mode_rx0_,
00128         const std::string &gain_mode_rx1_,
00129         double rf_gain_rx0_,
00130         double rf_gain_rx1_,
00131         bool enable_ch0,
00132         bool enable_ch1,
00133         int64_t freq_2ch,
00134         bool ppsmode_,
00135         bool customsamplesize_,
00136         const std::string &fe_ip_,
00137         int fe_ctlport_,
00138         int ssize_,
00139         int bshift_,
00140         bool spattern_,
00141         double lo_attenuation_db_,
00142         bool high_side_lo_,
00143         int tx_lo_channel_);
00144
00145     void ad9361_channel_demux_to_buffer(ad936x_iio_samples *samples_in, int nchannels,
      gr_vector_void_star &output_items);
00146     void ad9361_channel_demux_and_record(ad936x_iio_samples *samples_in, int nchannels,
      std::vector<std::fstream> *files_out);
00147
00148     std::thread pps_rx_thread;
00149     std::unique_ptr<ad936x_iio_custom> ad936x_custom;
00150     std::shared_ptr<pps_tcp_rx> pps_rx;
00151     std::shared_ptr<Concurrent_Queue<PpsSamplestamp» ppsqueue;
00152
00153     std::vector<std::fstream> samplesfile;
00154 };
00155
00156
00157 /** \} */
```

```
00158 /** \} */
00159 #endif  // GNSS_SDR_AD936X_IIO_SOURCE_H
```

13.368 bladerf_source.h File Reference

GNU Radio source block for Nuand's bladeRF front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), implemented directly against libbladeRF.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <cstdint>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class bladerf_source

  This class implements a GNU Radio source block that streams RX samples from a bladeRF device (single RX channel) using libbladeRF's synchronous streaming API, converting the device's native BLADERF_FORMAT_SC16_Q11 samples to gr_complex.

Typedefs

- using bladerf_source_sptr = gnss_shared_ptr<bladerf_source>

Functions

- bladerf_source_sptr **bladerf_make_source_sptr** (const std::string &bladerf_args, const std::string &device_serial, double sample_rate, double freq, double bandwidth, double gain, bool agc_enabled, bool rx_bias_tee, unsigned int num_buffers, unsigned int buffer_size, unsigned int num_transfers, unsigned int stream_timeout_ms)

13.368.1 Detailed Description

GNU Radio source block for Nuand's bladeRF front-ends (bladeRF x40, x115, and bladeRF 2.0 Micro xA4/xA9), implemented directly against libbladeRF.

Author

Oleksandr Suvorov, 2026.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file bladerf_source.h.

13.369 bladerf_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file bladerf_source.h
00003  * \brief GNU Radio source block for Nuand's bladeRF front-ends (bladeRF x40,
00004  * x115, and bladeRF 2.0 Micro xA4/xA9), implemented directly against
00005  * libbladeRF.
00006  * \author Oleksandr Suvorov, 2026.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
```

```
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_BLADERF_SOURCE_H
00020 #define GNSS_SDR_BLADERF_SOURCE_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/sync_block.h>
00024 #include <cstdint>
00025 #include <memory>
00026 #include <string>
00027 #include <vector>
00028
00029 /** \addtogroup Signal_Source
00030  * \{ */
00031 /** \addtogroup Signal_Source_gnuradio_blocks
00032  * \{ */
00033
00034
00035 class bladerf_source;
00036
00037 using bladerf_source_sptr = gnss_shared_ptr<bladerf_source>;
00038
00039 bladerf_source_sptr bladerf_make_source_sptr(
00040     const std::string &bladerf_args,
00041     const std::string &device_serial,
00042     double sample_rate,
00043     double freq,
00044     double bandwidth,
00045     double gain,
00046     bool agc_enabled,
00047     bool rx_bias_tee,
00048     unsigned int num_buffers,
00049     unsigned int buffer_size,
00050     unsigned int num_transfers,
00051     unsigned int stream_timeout_ms);
00052
00053 /*!
00054  * \brief This class implements a GNU Radio source block that streams RX
00055  * samples from a bladeRF device (single RX channel) using libbladeRF's
00056  * synchronous streaming API, converting the device's native
00057  * BLADERF_FORMAT_SC16_Q11 samples to gr_complex.
00058  */
00059 class bladerf_source : public gr::sync_block
00060 {
00061 public:
00062     ~bladerf_source() override = default;
00063
00064     bool start() override;
00065     bool stop() override;
00066
00067     int work(int noutput_items,
00068         gr_vector_const_void_star &input_items,
00069         gr_vector_void_star &output_items) override;
00070
00071 private:
00072     friend bladerf_source_sptr bladerf_make_source_sptr(
00073         const std::string &bladerf_args,
00074         const std::string &device_serial,
00075         double sample_rate,
00076         double freq,
00077         double bandwidth,
00078         double gain,
00079         bool agc_enabled,
00080         bool rx_bias_tee,
00081         unsigned int num_buffers,
00082         unsigned int buffer_size,
00083         unsigned int num_transfers,
00084         unsigned int stream_timeout_ms);
00085
00086     bladerf_source(
00087         const std::string &bladerf_args,
00088         const std::string &device_serial,
00089         double sample_rate,
00090         double freq,
00091         double bandwidth,
00092         double gain,
00093         bool agc_enabled,
00094         bool rx_bias_tee,
00095         unsigned int num_buffers,
00096         unsigned int buffer_size,
00097         unsigned int num_transfers,
00098         unsigned int stream_timeout_ms);
00099
00100     struct bladerf_deleter
00101     {
00102         void operator()(struct bladerf *dev) const;
```

```
00103     };
00104
00105     std::unique_ptr<struct bladerf, bladerf_deleter> dev_;
00106     std::vector<int16_t> raw_buffer_;
00107     unsigned int buffer_size_;
00108     unsigned int stream_timeout_ms_;
00109     bool streaming_;
00110 };
00111
00112
00113 /** \} */
00114 /** \} */
00115 #endif  // GNSS_SDR_BLADERF_SOURCE_H
```

13.370 evk1029_source.h File Reference

Unpacks the SAPHYRION EVK1029 raw capture files (continuous stream of OBA-encoded 4-bit samples, 16 samples packed per little-endian 64-bit word, no periodic block header).

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <fstream>
#include <string>
#include <vector>
```

Classes

- class Evk1029Source

  Reads a continuous, header-less EVK1029 raw capture file and unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first).

Typedefs

- using Evk1029Source_sptr = gnss_shared_ptr<Evk1029Source>

Functions

- Evk1029Source_sptr **evk1029_make_source** (const std::string &filename, Concurrent_Queue< pmt::pmt↩ _t > *queue, double sampling_frequency, uint64_t bytes_to_skip=0)

13.370.1 Detailed Description

Unpacks the SAPHYRION EVK1029 raw capture files (continuous stream of OBA-encoded 4-bit samples, 16 samples packed per little-endian 64-bit word, no periodic block header).
This is the successor of the old EVK10x9_source block, updated for the current EVK1029 capture format: the old format packed two 3-bit OBA samples per byte and inserted an 8-byte header every 2048-byte block; the current format is a continuous stream with no headers, packing two 4-bit OBA samples per byte (16 samples per 64-bit word, per the capture's .sdrx metadata: quantization=4, packedbits=64, encoding=OBA).
The block has a single output port. Multiple RF bands sharing the same raw capture (e.g. several Freq_Xlating_↩ Fir_Filter instances tuned to different IFs) are each connected to that one port (see the signal-source-to-conditioner wiring in gnss_flowgraph.cc, which takes this path whenever output_signature()->max_streams() == 1): GNU Radio supports several readers on one producer port, each with its own read pointer over the same buffer. The scheduler still throttles this block to the slowest of those readers (buffer::space_available() is bounded by the reader with the most unconsumed items), exactly as it would with one buffer per port, so the bands cannot drift apart by more than one output buffer. What the single port saves is the N-1 extra output buffers and the N-1 full-buffer memcpy calls per work() call, at up to the raw capture rate, that a port-per-band design needs in order to duplicate the same decoded samples.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file evk1029_source.h.

13.371 evk1029_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file evk1029_source.h
00003  * \brief Unpacks the SAPHYRION EVK1029 raw capture files (continuous stream
00004  * of OBA-encoded 4-bit samples, 16 samples packed per little-endian 64-bit
00005  * word, no periodic block header).
00006  *
00007  * This is the successor of the old EVK10x9_source block, updated for the
00008  * current EVK1029 capture format: the old format packed two 3-bit OBA
00009  * samples per byte and inserted an 8-byte header every 2048-byte block;
00010  * the current format is a continuous stream with no headers, packing two
00011  * 4-bit OBA samples per byte (16 samples per 64-bit word, per the
00012  * capture's .sdrx metadata: quantization=4, packedbits=64, encoding=OBA).
00013  *
00014  * The block has a single output port. Multiple RF bands sharing the same
00015  * raw capture (e.g. several Freq_Xlating_Fir_Filter instances tuned to
00016  * different IFs) are each connected to that one port (see the
00017  * signal-source-to-conditioner wiring in gnss_flowgraph.cc, which takes
00018  * this path whenever output_signature()->max_streams() == 1): GNU Radio
00019  * supports several readers on one producer port, each with its own read
00020  * pointer over the same buffer. The scheduler still throttles this block
00021  * to the slowest of those readers (buffer::space_available() is bounded by
00022  * the reader with the most unconsumed items), exactly as it would with one
00023  * buffer per port, so the bands cannot drift apart by more than one output
00024  * buffer. What the single port saves is the N-1 extra output buffers and
00025  * the N-1 full-buffer memcpy calls per work() call, at up to the raw
00026  * capture rate, that a port-per-band design needs in order to duplicate
00027  * the same decoded samples.
00028  *
00029  * -----------------------------------------------------------------------------
00030  *
00031  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00032  * This file is part of GNSS-SDR.
00033  *
00034  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00035  * SPDX-License-Identifier: GPL-3.0-or-later
00036  *
00037  * -----------------------------------------------------------------------------
00038  */
00039
00040 #ifndef GNSS_SDR_EVK1029_SOURCE_H
00041 #define GNSS_SDR_EVK1029_SOURCE_H
00042
00043 #include "concurrent_queue.h"
00044 #include "gnss_block_interface.h"
00045 #include <gnuradio/sync_block.h>
00046 #include <pmt/pmt.h>
00047 #include <cstdint>
00048 #include <fstream>
00049 #include <string>
00050 #include <vector>
00051
00052 /** \addtogroup Signal_Source
00053  * \{ */
00054 /** \addtogroup Signal_Source_gnuradio_blocks
00055  * \{ */
00056
00057
00058 class Evk1029Source;
00059
00060 using Evk1029Source_sptr = gnss_shared_ptr<Evk1029Source>;
00061
00062 // bytes_to_skip: byte position in the raw capture at which reading starts,
00063 // so that a long capture can be jumped straight to a time of interest
00064 // instead of being processed from the beginning (the adapter derives it
00065 // from its .seconds_to_skip parameter and the raw sampling frequency).
00066 // Rounded up to the next multiple of 8 bytes (64 bits) here, matching the
00067 // capture's native packing (16 OBA samples per little-endian 64-bit word):
00068 // this block itself only ever reads byte-by-byte and would decode correctly
00069 // from any byte offset, but seeking to a non-word-aligned byte splits a
00070 // 64-bit DMA word the real hardware always transferred as one unit, which
00071 // needlessly complicates comparing a jumped-to run against a from-the-start
00072 // one at the file/word level.
00073 Evk1029Source_sptr evk1029_make_source(const std::string& filename, Concurrent_Queue<pmt::pmt_t>*
      queue, double sampling_frequency, uint64_t bytes_to_skip = 0);
00074
00075 /*!
00076  * \brief Reads a continuous, header-less EVK1029 raw capture file and
00077  * unpacks its OBA-encoded 4-bit samples (two per byte, low nibble first).
00078  */
00079 class Evk1029Source : public gr::sync_block
00080 {
00081 public:
00082     ~Evk1029Source() override;
```

```
00083
00084 private:
00085     friend Evk1029Source_sptr evk1029_make_source(const std::string& filename,
      Concurrent_Queue<pmt::pmt_t>* queue, double sampling_frequency, uint64_t bytes_to_skip);
00086     Evk1029Source(const std::string& filename, Concurrent_Queue<pmt::pmt_t>* queue, double
      sampling_frequency, uint64_t bytes_to_skip = 0);
00087
00088     int work(int noutput_items,
00089         gr_vector_const_void_star& input_items,
00090         gr_vector_void_star& output_items) override;
00091
00092     std::ifstream binary_input_file_;
00093     Concurrent_Queue<pmt::pmt_t>* queue_;
00094     std::vector<uint8_t> buffer_;
00095     std::size_t buffer_valid_;  // number of valid bytes currently in buffer_
00096     std::size_t buffer_pos_;    // next unread byte offset within buffer_
00097 };
00098
00099
00100 /** \} */
00101 /** \} */
00102 #endif  // GNSS_SDR_EVK1029_SOURCE_H
```

13.372 fifo_reader.h File Reference

Header file to retrieve samples from an existing Unix FIFO.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <array>
#include <fstream>
#include <string>
```

Classes

- class FifoReader

13.372.1 Detailed Description

Header file to retrieve samples from an existing Unix FIFO.

Author

Malte Lenhart, 2021. malte.lenhart(at)mailbox.org

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fifo_reader.h.

13.373 fifo_reader.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fifo_reader.h
00003  *
00004  * \brief Header file to retrieve samples from an existing Unix FIFO
00005  * \author Malte Lenhart, 2021. malte.lenhart(at)mailbox.org
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_FIFO_READER_H_
00019 #define GNSS_SDR_FIFO_READER_H_
00020
```

```
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_block.h>
00023 #include <array>
00024 #include <fstream>  // std::ifstream
00025 #include <string>
00026
00027 /** \addtogroup Signal_Source
00028  * \{ */
00029 /** \addtogroup Signal_Source_gnuradio_blocks
00030  * \{ */
00031 class FifoReader : virtual public gr::sync_block
00032 {
00033 public:
00034     //! \brief static function to create a class instance
00035     using sptr = gnss_shared_ptr<FifoReader>;
00036     static sptr make(const std::string &file_name, const std::string &sample_type);
00037
00038     ~FifoReader() = default;
00039
00040     //! initialize istream resource for FIFO
00041     bool start();
00042
00043     // gnu radio work cycle function
00044     int work(int noutput_items,
00045         gr_vector_const_void_star &input_items,
00046         gr_vector_void_star &output_items);
00047
00048 private:
00049     //! \brief Constructor
00050     //! private constructor called by function make
00051     //! (gr handles this with public and private header pair)
00052     FifoReader(const std::string &file_name, const std::string &sample_type);
00053
00054     size_t read_gr_complex(int noutput_items, gr_vector_void_star &output_items);
00055
00056     //! function to read data out of FIFO which is stored as interleaved I/Q stream.
00057     //! template argument determines sample_type
00058     // Note: template definition necessary in header file
00059     // See also:
      https://stackoverflow.com/questions/495021/why-can-templates-only-be-implemented-in-the-header-file
00060     template <typename Type>
00061     size_t read_interleaved(int noutput_items, gr_vector_void_star &output_items)
00062     {
00063         size_t items_retrieved = 0;
00064         for (int n = 0; n < noutput_items; n++)
00065             {
00066                 // TODO: try if performance increases if we copy larger chunks to vector.
00067                 // how to read from stream: https://en.cppreference.com/w/cpp/io/basic_ifstream
00068                 std::array<char, 2 * sizeof(Type)> buffer;
00069                 fifo_.read(reinterpret_cast<char *>(buffer.data()), buffer.size());
00070                 if (fifo_.good())
00071                     {
00072                         auto real = reinterpret_cast<Type const *>(&buffer[0]);
00073                         auto imag = reinterpret_cast<Type const *>(&buffer[sizeof(Type)]);
00074                         static_cast<gr_complex *>(output_items[0])[n] = gr_complex(*real, *imag);
00075                         items_retrieved++;
00076                     }
00077                 else if (fifo_.eof())
00078                     {
00079                         fifo_.clear();
00080                         break;
00081                     }
00082                 else
00083                     {
00084                         fifo_error_output();
00085                         break;
00086                     }
00087             }
00088         return items_retrieved;
00089     }
00090
00091     //! this function moves logging output from this header into the source file
00092     //! thereby eliminating the need to include glog/logging.h in this header
00093     void fifo_error_output() const;
00094
00095     const std::string file_name_;
00096     const std::string sample_type_;
00097     std::ifstream fifo_;
00098 };
00099
00100 /** \} */
00101 /** \} */
00102 #endif /* GNSS_SDR_FIFO_READER_H_ */
```

13.374 gr_complex_ip_packet_source.h File Reference

Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap).

```
#include "gnss_block_interface.h"
#include <boost/thread.hpp>
#include <gnuradio/sync_block.h>
#include <arpa/inet.h>
#include <net/ethernet.h>
#include <net/if.h>
#include <netinet/if_ether.h>
#include <pcap.h>
#include <string>
#include <sys/ioctl.h>
```

Classes

- class Gr_Complex_Ip_Packet_Source

13.374.1 Detailed Description

Receives ip frames containing samples in UDP frame encapsulation using a high performance packet capture library (libpcap).

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gr_complex_ip_packet_source.h.

13.375 gr_complex_ip_packet_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gr_complex_ip_packet_source.h
00003  *
00004  * \brief Receives ip frames containing samples in UDP frame encapsulation
00005  * using a high performance packet capture library (libpcap)
00006  * \author Javier Arribas jarribas (at) cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_GR_COMPLEX_IP_PACKET_SOURCE_H
00021 #define GNSS_SDR_GR_COMPLEX_IP_PACKET_SOURCE_H
00022
00023 #include "gnss_block_interface.h"
00024 #include <boost/thread.hpp>
00025 #include <gnuradio/sync_block.h>
00026 #include <arpa/inet.h>
00027 #include <net/ethernet.h>
00028 #include <net/if.h>
00029 #include <netinet/if_ether.h>
00030 #include <pcap.h>
00031 #include <string>
00032 #include <sys/ioctl.h>
00033
00034 /** \addtogroup Signal_Source
00035  * \{ */
```

```
00036 /** \addtogroup Signal_Source_gnuradio_blocks signal_source_gr_blocks
00037  * GNU Radio blocks for signal sources.
00038  * \{ */
00039
00040
00041 class Gr_Complex_Ip_Packet_Source : virtual public gr::sync_block
00042 {
00043 public:
00044     using sptr = gnss_shared_ptr<Gr_Complex_Ip_Packet_Source>;
00045     static sptr make(std::string src_device,
00046         const std::string &origin_address,
00047         int udp_port,
00048         int udp_packet_size,
00049         int n_baseband_channels,
00050         const std::string &wire_sample_type,
00051         size_t item_size,
00052         bool IQ_swap_);
00053     Gr_Complex_Ip_Packet_Source(std::string src_device,
00054         const std::string &origin_address,
00055         int udp_port,
00056         int udp_packet_size,
00057         int n_baseband_channels,
00058         const std::string &wire_sample_type,
00059         size_t item_size,
00060         bool IQ_swap_);
00061     ~Gr_Complex_Ip_Packet_Source();
00062
00063     // Called by gnuradio to enable drivers, etc for i/o devices.
00064     bool start();
00065
00066     // Called by gnuradio to disable drivers, etc for i/o devices.
00067     bool stop();
00068
00069     // Where all the action really happens
00070     int work(int noutput_items,
00071         gr_vector_const_void_star &input_items,
00072         gr_vector_void_star &output_items);
00073
00074 private:
00075     void demux_samples(const gr_vector_void_star &output_items, int num_samples_readed);
00076     void my_pcap_loop_thread(pcap_t *pcap_handle);
00077     void pcap_callback(u_char *args, const struct pcap_pkthdr *pkthdr, const u_char *packet);
00078     static void static_pcap_callback(u_char *args, const struct pcap_pkthdr *pkthdr, const u_char
      *packet);
00079     /*
00080      * Opens the ethernet device using libpcap raw capture mode
00081      * If any of these fail, the function returns the error and exits.
00082      */
00083     bool open();
00084
00085     boost::thread *d_pcap_thread;
00086     // boost::mutex d_mutex;
00087     struct sockaddr_in si_me{};
00088     std::string d_src_device;
00089     std::string d_origin_address;
00090     pcap_t *descr;  // ethernet pcap device descriptor
00091     char *fifo_buff;
00092     int fifo_read_ptr;
00093     int fifo_write_ptr;
00094     int fifo_items;
00095     int d_sock_raw;
00096     int d_udp_port;
00097     int d_n_baseband_channels;
00098     int d_wire_sample_type;
00099     float d_bytes_per_sample;
00100     bool d_IQ_swap;
00101 };
00102
00103
00104 /** \} */
00105 /** \} */
00106 #endif  //  GNSS_SDR_GR_COMPLEX_IP_PACKET_SOURCE_H
```

13.376 ion_gsms.h File Reference

GNU Radio block that reads a Block from a file following ION's GNSS-SDR metadata standard.

```
#include "gnss_block_interface.h"
#include "gnss_sdr_filesystem.h"
#include "ion_gsms_chunk_data.h"
#include <gnuradio/block.h>
#include <gnuradio/sync_block.h>
```

```
#include <cstddef>
#include <fstream>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class IONGSMSFileSource
- struct IONGSMSFileSource::SegmentDescriptor

13.376.1 Detailed Description

GNU Radio block that reads a Block from a file following ION's GNSS-SDR metadata standard.

Author

Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ion_gsms.h.

13.377 ion_gsms.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ion_gsms.h
00003  * \brief GNU Radio block that reads a Block from a file following ION's GNSS-SDR metadata standard
00004  * \author Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com
00005  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_ION_GSMS_H
00019 #define GNSS_SDR_ION_GSMS_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "gnss_sdr_filesystem.h"
00023 #include "ion_gsms_chunk_data.h"
00024 #include <gnuradio/block.h>
00025 #include <gnuradio/sync_block.h>
00026 #include <cstddef>
00027 #include <fstream>
00028 #include <memory>
00029 #include <string>
00030 #include <vector>
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_gnuradio_blocks
00035  * \{ */
00036
00037 class IONGSMSFileSource : public gr::sync_block
00038 {
00039 public:
00040     using sptr = gnss_shared_ptr<IONGSMSFileSource>;
00041
00042     struct SegmentDescriptor
00043     {
00044         fs::path data_filepath;
00045         const GnssMetadata::Block* block = nullptr;
00046         std::size_t block_start_offset = 0;
```

```
00047         bool block_extends_to_eof = false;
00048     };
00049
00050     IONGSMSFileSource(
00051         const fs::path& metadata_filepath,
00052         const GnssMetadata::File& file,
00053         const GnssMetadata::Block& block,
00054         std::size_t block_start_offset,
00055         const std::vector<std::string>& stream_ids);
00056
00057     IONGSMSFileSource(
00058         const std::vector<SegmentDescriptor>& segments,
00059         const std::vector<std::string>& stream_ids);
00060
00061     int work(
00062         int noutput_items,
00063         gr_vector_const_void_star& input_items,
00064         gr_vector_void_star& output_items) override;
00065
00066     std::size_t output_stream_count() const;
00067     std::size_t output_stream_item_size(std::size_t stream_index) const;
00068     std::size_t output_stream_total_sample_count(std::size_t stream_index) const;
00069
00070 private:
00071     struct SegmentData
00072     {
00073         fs::path data_filepath;
00074         const GnssMetadata::Block* block = nullptr;
00075         std::size_t block_start_offset = 0;
00076         std::size_t cycle_count = 0;
00077         std::size_t chunk_cycle_length = 0;
00078         std::size_t maximum_item_rate = 0;
00079         std::vector<std::size_t> output_stream_item_rates;
00080         std::vector<std::shared_ptr<IONGSMSChunkData» chunk_data;
00081     };
00082
00083     static SegmentDescriptor make_segment_descriptor(
00084         const fs::path& metadata_filepath,
00085         const GnssMetadata::File& file,
00086         const GnssMetadata::Block& block,
00087         std::size_t block_start_offset);
00088     static gr::io_signature::sptr make_output_signature(const std::vector<SegmentDescriptor>&
      segments, const std::vector<std::string>& stream_ids);
00089     static bool block_contains_stream_id(const GnssMetadata::Block& block, const std::string&
      stream_id);
00090     static std::vector<std::string> block_output_stream_ids(const GnssMetadata::Block& block, const
      std::vector<std::string>& stream_ids);
00091     static std::vector<std::string> segment_output_stream_ids(const std::vector<SegmentDescriptor>&
      segments, const std::vector<std::string>& stream_ids);
00092     static int output_item_size_for_stream_id(const GnssMetadata::Block& block, const std::string&
      stream_id);
00093     static int output_item_size_for_stream_id(const std::vector<SegmentDescriptor>& segments, const
      std::string& stream_id);
00094     static std::size_t infer_cycle_count_from_file(
00095         const fs::path& data_filepath,
00096         const GnssMetadata::Block& block,
00097         std::size_t block_start_offset,
00098         std::size_t chunk_cycle_length);
00099     bool advance_to_next_segment();
00100
00101     std::ifstream file_stream_;
00102     std::vector<char> io_buffer_;
00103     std::size_t io_buffer_offset_;
00104     std::vector<int> items_produced_;
00105     std::size_t output_stream_count_;
00106     std::vector<std::size_t> output_stream_item_sizes_;
00107     std::vector<std::size_t> output_stream_item_rates_;
00108     std::vector<std::size_t> output_stream_total_sample_counts_;
00109     std::size_t maximum_item_rate_;
00110     std::size_t chunk_cycle_length_;
00111     std::size_t cycles_remaining_;
00112     std::vector<SegmentData> segments_;
00113     std::size_t current_segment_index_;
00114     std::vector<std::string> output_stream_ids_;
00115 };
00116
00117 /** \} */
00118 /** \} */
00119 #endif  // GNSS_SDR_ION_GSMS_H
```

13.378 labsat23_source.h File Reference

Unpacks capture files in the LabSat 2 (ls2), LabSat 3 (ls3), LabSat 3 Wideband (LS3W), and Labsat 4 (ls4) formats.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <bitset>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <map>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class labsat23_source

  This class implements conversion between Labsat 2, 3 and 3 Wideband formats to gr_complex.

Typedefs

- using labsat23_source_sptr = gnss_shared_ptr<labsat23_source>

Functions

- labsat23_source_sptr **labsat23_make_source_sptr** (const char *signal_file_basename, const std::vector< int > &channel_selector, Concurrent_Queue< pmt::pmt_t > *queue, bool digital_io_enabled, double seconds_to_skip)

13.378.1 Detailed Description

Unpacks capture files in the LabSat 2 (ls2), LabSat 3 (ls3), LabSat 3 Wideband (LS3W), and Labsat 4 (ls4) formats.

Author

Javier Arribas jarribas (at) cttc.es Mathieu Favreau favreau.mathieu (at) hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file labsat23_source.h.

13.379 labsat23_source.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file labsat23_source.h
00003  *
00004  * \brief Unpacks capture files in the LabSat 2 (ls2), LabSat 3 (ls3), LabSat 3
00005  * Wideband (LS3W), and Labsat 4 (ls4) formats.
00006  * \author Javier Arribas jarribas (at) cttc.es
00007  *         Mathieu Favreau favreau.mathieu (at) hotmail.com
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_LABSAT23_SOURCE_H
00021 #define GNSS_SDR_LABSAT23_SOURCE_H
00022
00023 #include "concurrent_queue.h"
```

```
00024 #include "gnss_block_interface.h"
00025 #include <gnuradio/block.h>
00026 #include <pmt/pmt.h>
00027 #include <bitset>
00028 #include <cstddef>
00029 #include <cstdint>
00030 #include <fstream>
00031 #include <map>
00032 #include <string>
00033 #include <utility>
00034 #include <vector>
00035
00036 /** \addtogroup Signal_Source
00037  * \{ */
00038 /** \addtogroup Signal_Source_gnuradio_blocks
00039  * \{ */
00040
00041
00042 class labsat23_source;
00043
00044 using labsat23_source_sptr = gnss_shared_ptr<labsat23_source>;
00045
00046 labsat23_source_sptr labsat23_make_source_sptr(
00047     const char *signal_file_basename,
00048     const std::vector<int> &channel_selector,
00049     Concurrent_Queue<pmt::pmt_t> *queue,
00050     bool digital_io_enabled,
00051     double seconds_to_skip);
00052
00053 /*!
00054  * \brief This class implements conversion between Labsat 2, 3 and 3 Wideband
00055  * formats to gr_complex
00056  */
00057 class labsat23_source : public gr::block
00058 {
00059 public:
00060     ~labsat23_source();
00061
00062     int general_work(int noutput_items,
00063         gr_vector_int &ninput_items,
00064         gr_vector_const_void_star &input_items,
00065         gr_vector_void_star &output_items);
00066
00067 private:
00068     friend labsat23_source_sptr labsat23_make_source_sptr(
00069         const char *signal_file_basename,
00070         const std::vector<int> &channel_selector,
00071         Concurrent_Queue<pmt::pmt_t> *queue,
00072         bool digital_io_enabled,
00073         double seconds_to_skip);
00074
00075     labsat23_source(const char *signal_file_basename,
00076         const std::vector<int> &channel_selector,
00077         Concurrent_Queue<pmt::pmt_t> *queue,
00078         bool digital_io_enabled,
00079         double seconds_to_skip);
00080
00081     static std::vector<double> generate_mapping(int qua);
00082     static void write_samples_from_bitset(const std::bitset<64> &bs, int bit_offset, int qua,
      gr_complex &out);
00083     static void write_samples_ls4(uint64_t data_index, int out_index, int sample_count, int32_t qua,
      std::vector<uint64_t> &data, gr_complex *out_samples);
00084     static void read_file_register_to_local_endian(std::ifstream &binary_input_file, uint64_t
      &read_register);
00085     static bool are_equal_ignore_nonpositive(const std::vector<int32_t> &values);
00086     static uint8_t as_u8(char value);
00087     static uint16_t read_le_u16(const char *data);
00088     static uint32_t read_le_u32(const char *data);
00089
00090     std::string generate_filename();
00091
00092     int parse_header();
00093     int read_ls3w_ini(const std::string &filename);
00094     int number_of_samples_per_ls3w_register() const;
00095     int configure_ls4_output_parameters();
00096
00097     void decode_samples_one_channel(uint16_t input_word, gr_complex *out, int type);
00098     void decode_samples_two_channels(uint16_t input_word, std::vector<gr_complex *> &out, std::size_t
      output_pointer) const;
00099     void decode_samples_three_channels(uint32_t input_word, std::vector<gr_complex *> &out,
      std::size_t output_pointer) const;
00100     void decode_ls3w_register(uint64_t input, std::vector<gr_complex *> &out, std::size_t
      output_pointer) const;
00101     int parse_ls23_data(int noutput_items, std::vector<gr_complex *> out);
00102     int parse_ls3w_data(int noutput_items, std::vector<gr_complex *> out);
00103     int parse_ls4_data(int noutput_items, std::vector<gr_complex *> out);
00104     bool read_ls4_data();
```

```
00105
00106     std::ifstream binary_input_file;
00107     std::string d_signal_file_basename;
00108     Concurrent_Queue<pmt::pmt_t> *d_queue;
00109     std::vector<int> d_channel_selector_config;
00110     int d_current_file_number;
00111     uint8_t d_labsat_version;
00112     uint8_t d_channel_selector;
00113     uint8_t d_ref_clock;
00114     uint8_t d_bits_per_sample;
00115     bool d_header_parsed;
00116
00117     // Data members for Labsat 3 Wideband and Labsat 4
00118     std::string d_ls3w_OSC;
00119     std::vector<int> d_ls3w_selected_channel_offset;
00120     int64_t d_ls3w_SMP{};
00121     int32_t d_sample_quantization_bits{};
00122     int32_t d_ls3w_CHN{};
00123     int32_t d_ls3w_SFT{};
00124     int d_ls3w_spare_bits{};
00125     int d_ls3w_samples_per_register{};
00126     bool d_is_ls3w = false;
00127     bool d_is_ls4 = false;
00128     bool d_ls3w_digital_io_enabled = false;
00129
00130     // Data members for Labsat 4
00131     int32_t d_ls4_BW_MAX{0};
00132     int32_t d_number_sample_per_output{0};
00133     int32_t d_number_register_per_output{0};
00134     uint64_t d_read_index{0};
00135
00136     struct ChannelState
00137     {
00138         std::string identifier;
00139         int32_t center_freq{0};
00140         int32_t bandwidth{0};
00141         int32_t bw_div{0};
00142         int32_t buff_size{0};
00143         int32_t number_sample_per_output{0};
00144         uint64_t data_index{0};
00145         std::vector<uint64_t> data{};
00146
00147         ChannelState(const std::string &id) : identifier(id) {}
00148     };
00149
00150     std::map<int32_t, ChannelState> d_channel_map{
00151         std::make_pair(1, ChannelState{"A"}), std::make_pair(2, ChannelState{"B"}), std::make_pair(3,
      ChannelState{"C"})};
00152 };
00153
00154
00155 /** \} */
00156 /** \} */
00157 #endif  // GNSS_SDR_LABSAT23_SOURCE_H
```

13.380 rtl_tcp_signal_source_c.h File Reference

Interface of an rtl_tcp signal source reader.

```
#include "gnss_block_interface.h"
#include "rtl_tcp_dongle_info.h"
#include <boost/array.hpp>
#include <boost/asio.hpp>
#include <boost/circular_buffer.hpp>
#include <boost/thread/condition.hpp>
#include <boost/thread/mutex.hpp>
#include <gnuradio/sync_block.h>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class rtl_tcp_signal_source_c

  This class reads interleaved I/Q samples from an rtl_tcp server and outputs complex types.

Typedefs

- using rtl_tcp_signal_source_c_sptr = gnss_shared_ptr<rtl_tcp_signal_source_c>
- using b_io_context = boost::asio::io_service

Functions

- rtl_tcp_signal_source_c_sptr **rtl_tcp_make_signal_source_c** (const std::string &address, int16_t port, bool flip_iq=false)

13.380.1 Detailed Description

Interface of an rtl_tcp signal source reader.

Author

Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au

The implementation of this block is a combination of various helpful sources. The data format and command structure is taken from the original Osmocom rtl_tcp_source_f (https://git.osmocom.org/gr-osmosdr). The asynchronous reading code comes from the examples provides by Boost.Asio and the bounded buffer producer-consumer solution is taken from the Boost.CircularBuffer examples (https://www.boost.org/).

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file rtl_tcp_signal_source_c.h.

13.381 rtl_tcp_signal_source_c.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtl_tcp_signal_source_c.h
00003  * \brief Interface of an rtl_tcp signal source reader.
00004  * \author Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au
00005  *
00006  * The implementation of this block is a combination of various helpful
00007  * sources. The data format and command structure is taken from the
00008  * original Osmocom rtl_tcp_source_f (https://git.osmocom.org/gr-osmosdr).
00009  * The asynchronous reading code comes from the examples provides
00010  * by Boost.Asio and the bounded buffer producer-consumer solution is
00011  * taken from the Boost.CircularBuffer examples (https://www.boost.org/).
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_C_H
00025 #define GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_C_H
00026
00027 #include "gnss_block_interface.h"
00028 #include "rtl_tcp_dongle_info.h"
00029 #include <boost/array.hpp>
00030 #include <boost/asio.hpp>
00031 #include <boost/circular_buffer.hpp>
00032 #include <boost/thread/condition.hpp>
00033 #include <boost/thread/mutex.hpp>
00034 #include <gnuradio/sync_block.h>
00035 #include <cstdint>
00036 #include <string>
00037 #include <vector>
00038
00039
00040 /** \addtogroup Signal_Source
00041  * \{ */
00042 /** \addtogroup Signal_Source_gnuradio_blocks
00043  * \{ */
00044
00045
00046 class rtl_tcp_signal_source_c;
```

```
00047
00048 using rtl_tcp_signal_source_c_sptr = gnss_shared_ptr<rtl_tcp_signal_source_c>;
00049
00050 #if USE_BOOST_ASIO_IO_CONTEXT
00051 using b_io_context = boost::asio::io_context;
00052 #else
00053 using b_io_context = boost::asio::io_service;
00054 #endif
00055
00056 rtl_tcp_signal_source_c_sptr
00057 rtl_tcp_make_signal_source_c(const std::string &address,
00058     int16_t port,
00059     bool flip_iq = false);
00060
00061 /*!
00062  * \brief This class reads interleaved I/Q samples
00063  * from an rtl_tcp server and outputs complex types.
00064  */
00065 class rtl_tcp_signal_source_c : public gr::sync_block
00066 {
00067 public:
00068     ~rtl_tcp_signal_source_c();
00069
00070     int work(int noutput_items,
00071         gr_vector_const_void_star &input_items,
00072         gr_vector_void_star &output_items);
00073
00074     void set_frequency(int frequency);
00075     void set_sample_rate(int sample_rate);
00076     void set_agc_mode(bool agc);
00077     void set_gain(int gain);
00078     void set_if_gain(int gain);
00079
00080 private:
00081     friend rtl_tcp_signal_source_c_sptr
00082     rtl_tcp_make_signal_source_c(const std::string &address,
00083         int16_t port,
00084         bool flip_iq);
00085
00086     rtl_tcp_signal_source_c(const std::string &address,
00087         int16_t port,
00088         bool flip_iq);
00089
00090     // async read callback
00091     void handle_read(const boost::system::error_code &ec,
00092         size_t bytes_transferred);
00093
00094     inline bool not_full() const
00095     {
00096         return unread_ < buffer_.capacity();
00097     }
00098
00099     inline bool not_empty() const
00100     {
00101         return unread_ > 0 || io_context_.stopped();
00102     }
00103
00104     boost::circular_buffer_space_optimized<float> buffer_;
00105     // producer-consumer helpers
00106     boost::mutex mutex_;
00107     boost::condition not_full_;
00108     boost::condition not_empty_;
00109
00110     // lookup for scaling data
00111     boost::array<float, 0xff> lookup_{};
00112
00113     // IO members
00114     b_io_context io_context_;
00115     boost::asio::ip::tcp::socket socket_;
00116     std::vector<unsigned char> data_;
00117
00118     Rtl_Tcp_Dongle_Info info_;
00119     size_t unread_;
00120     bool flip_iq_;
00121 };
00122
00123
00124 /** \} */
00125 /** \} */
00126 #endif  // GNSS_SDR_RTL_TCP_SIGNAL_SOURCE_C_H
```

13.382 unpack_2bit_samples.h File Reference

Unpacks 2 bit samples samples may be packed in any of the following ways: 1) Into bytes [item == byte] 1a) Big endian ordering within the byte 1b) Little endian ordering within the byte 2) Into shorts [item == short] 2a) Big endian ordering of bytes, big endian within the byte 2b) Big endian ordering of bytes, little endian within the byte 2c) Little endian ordering of bytes, big endian within the byte 2d) Little endian ordering of bytes, little endian within the byte.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
#include <cstdint>
#include <vector>
```

Classes

- class unpack_2bit_samples

  This class takes 2 bit samples that have been packed into bytes or shorts as input and generates a byte for each sample. It generates eight times as much data as is input (every two bits become 16 bits).

Typedefs

- using unpack_2bit_samples_sptr = gnss_shared_ptr<unpack_2bit_samples>

Functions

- unpack_2bit_samples_sptr **make_unpack_2bit_samples** (bool big_endian_bytes, size_t item_size, bool big_endian_items, bool reverse_interleaving=false)

13.382.1 Detailed Description

Unpacks 2 bit samples samples may be packed in any of the following ways: 1) Into bytes [item == byte] 1a) Big endian ordering within the byte 1b) Little endian ordering within the byte 2) Into shorts [item == short] 2a) Big endian ordering of bytes, big endian within the byte 2b) Big endian ordering of bytes, little endian within the byte 2c) Little endian ordering of bytes, big endian within the byte 2d) Little endian ordering of bytes, little endian within the byte.
Within a byte the two possibilities look like this: 7 6 5 4 3 2 1 0 : Bit number x_n,1 x_n,0 x_n+1,1 x_n+1,0 x_n+2,1 x_n+2,0 x_n+3,1 x_n+3,0 : Little endian x_n+3,1 x_n+3,0 x_n+2,1 x_n+2,0 x_n+1,0 x_n+1,0 x_n,1 x_n, 0 : Big Endian
For a short (uint16_t) the bytes are either transmitted as follows:
1 0 : Byte number Byte_n Byte_n+1 : Little endian Byte_n+1 Byte_n : Bit endian
The two bit values are assumed to have the following mapping:
x_1 x_0 Value 0 0 +1 0 1 +3 1 0 -3 1 1 -1
Letting x denote the two's complement interpretation of x_1 x_0, then:
Value = 2∗x + 1
We want to output the data in the order:
Value_0, Value_1, Value_2, ..., Value_n, Value_n+1, Value_n+2, ...

Author

Cillian O'Driscoll cillian.odriscoll (at) gmail . com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file unpack_2bit_samples.h.

13.383 unpack_2bit_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_2bit_samples.h
```

```
00003  *
00004  * \brief Unpacks 2 bit samples
00005  *  samples may be packed in any of the following ways:
00006  *  1) Into bytes [ item == byte ]
00007  *      1a) Big endian ordering within the byte
00008  *      1b) Little endian ordering within the byte
00009  *  2) Into shorts [ item == short ]
00010  *      2a) Big endian ordering of bytes, big endian within the byte
00011  *      2b) Big endian ordering of bytes, little endian within the byte
00012  *      2c) Little endian ordering of bytes, big endian within the byte
00013  *      2d) Little endian ordering of bytes, little endian within the byte
00014  *
00015  *  Within a byte the two possibilities look like this:
00016  *    7       6       5       4       3       2       1       0     : Bit number
00017  *  x_n,1   x_n,0  x_n+1,1 x_n+1,0 x_n+2,1 x_n+2,0 x_n+3,1 x_n+3,0  : Little endian
00018  * x_n+3,1 x_n+3,0 x_n+2,1 x_n+2,0 x_n+1,0 x_n+1,0  x_n,1   x_n, 0  : Big Endian
00019  *
00020  *  For a short (uint16_t) the bytes are either transmitted as follows:
00021  *
00022  *   1       0         : Byte number
00023  * Byte_n   Byte_n+1   : Little endian
00024  * Byte_n+1 Byte_n     : Bit endian
00025  *
00026  *  The two bit values are assumed to have the following mapping:
00027  *
00028  *  x_1  x_0     Value
00029  *   0    0       +1
00030  *   0    1       +3
00031  *   1    0       -3
00032  *   1    1       -1
00033  *
00034  *   Letting x denote the two's complement interpretation of x_1 x_0, then:
00035  *
00036  *   Value = 2*x + 1
00037  *
00038  *   We want to output the data in the order:
00039  *
00040  *   Value_0, Value_1, Value_2, ..., Value_n, Value_n+1, Value_n+2, ...
00041  *
00042  * \author Cillian O'Driscoll cillian.odriscoll (at) gmail . com
00043  *
00044  * -----------------------------------------------------------------------------
00045  *
00046  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00047  * This file is part of GNSS-SDR.
00048  *
00049  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00050  * SPDX-License-Identifier: GPL-3.0-or-later
00051  *
00052  * -----------------------------------------------------------------------------
00053  */
00054
00055 #ifndef GNSS_SDR_UNPACK_2BIT_SAMPLES_H
00056 #define GNSS_SDR_UNPACK_2BIT_SAMPLES_H
00057
00058 #include "gnss_block_interface.h"
00059 #include <gnuradio/sync_interpolator.h>
00060 #include <cstdint>
00061 #include <vector>
00062
00063 /** \addtogroup Signal_Source
00064  * \{ */
00065 /** \addtogroup Signal_Source_gnuradio_blocks
00066  * \{ */
00067
00068
00069 class unpack_2bit_samples;
00070
00071 using unpack_2bit_samples_sptr = gnss_shared_ptr<unpack_2bit_samples>;
00072
00073 unpack_2bit_samples_sptr make_unpack_2bit_samples(
00074     bool big_endian_bytes,
00075     size_t item_size,
00076     bool big_endian_items,
00077     bool reverse_interleaving = false);
00078
00079 /*!
00080  * \brief This class takes 2 bit samples that have been packed into bytes or
00081  * shorts as input and generates a byte for each sample. It generates eight
00082  * times as much data as is input (every two bits become 16 bits)
00083  */
00084 class unpack_2bit_samples : public gr::sync_interpolator
00085 {
00086 public:
00087     ~unpack_2bit_samples() = default;
00088
00089     unpack_2bit_samples(bool big_endian_bytes,
```

```
00090         size_t item_size,
00091         bool big_endian_items,
00092         bool reverse_interleaving);
00093
00094     int work(int noutput_items,
00095         gr_vector_const_void_star &input_items,
00096         gr_vector_void_star &output_items);
00097
00098 private:
00099     friend unpack_2bit_samples_sptr make_unpack_2bit_samples_sptr(
00100         bool big_endian_bytes,
00101         size_t item_size,
00102         bool big_endian_items,
00103         bool reverse_interleaving);
00104
00105     std::vector<int8_t> work_buffer_;
00106     size_t item_size_;
00107     bool big_endian_bytes_;
00108     bool big_endian_items_;
00109     bool swap_endian_items_;
00110     bool swap_endian_bytes_;
00111     bool reverse_interleaving_;
00112 };
00113
00114
00115 /** \} */
00116 /** \} */
00117 #endif  // GNSS_SDR_UNPACK_2BIT_SAMPLES_H
```

13.384 unpack_byte_2bit_cpx_samples.h File Reference

Unpacks byte samples to 2 bits complex samples. Packing Order Most Significant Nibble - Sample n Least Significant Nibble - Sample n+1 Packing order in Nibble Q1 Q0 I1 I0.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
```

Classes

- class unpack_byte_2bit_cpx_samples

  This class implements conversion between byte packet samples to 2bit_cpx samples 1 byte = 2 x complex 2bit I, + 2bit Q samples.

Typedefs

- using unpack_byte_2bit_cpx_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_cpx_samples>

Functions

- unpack_byte_2bit_cpx_samples_sptr **make_unpack_byte_2bit_cpx_samples** ()

13.384.1 Detailed Description

Unpacks byte samples to 2 bits complex samples. Packing Order Most Significant Nibble - Sample n Least Significant Nibble - Sample n+1 Packing order in Nibble Q1 Q0 I1 I0.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file unpack_byte_2bit_cpx_samples.h.

13.385 unpack_byte_2bit_cpx_samples.h

Go to the documentation of this file.

```
00001 /*!
```

```
00002  * \file unpack_byte_2bit_cpx_samples.h
00003  *
00004  * \brief Unpacks byte samples to 2 bits complex samples.
00005  *     Packing Order
00006  *     Most Significant Nibble  - Sample n
00007  *     Least Significant Nibble - Sample n+1
00008  *     Packing order in Nibble Q1 Q0 I1 I0
00009  * \author Javier Arribas jarribas (at) cttc.es
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_UNPACK_BYTE_2BIT_CPX_SAMPLES_H
00023 #define GNSS_SDR_UNPACK_BYTE_2BIT_CPX_SAMPLES_H
00024
00025 #include "gnss_block_interface.h"
00026 #include <gnuradio/sync_interpolator.h>
00027
00028 /** \addtogroup Signal_Source
00029  * \{ */
00030 /** \addtogroup Signal_Source_gnuradio_blocks
00031  * \{ */
00032
00033
00034 class unpack_byte_2bit_cpx_samples;
00035
00036 using unpack_byte_2bit_cpx_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_cpx_samples>;
00037
00038 unpack_byte_2bit_cpx_samples_sptr make_unpack_byte_2bit_cpx_samples();
00039
00040 /*!
00041  * \brief This class implements conversion between byte packet samples to 2bit_cpx samples
00042  *  1 byte = 2 x complex 2bit I, + 2bit Q samples
00043  */
00044 class unpack_byte_2bit_cpx_samples : public gr::sync_interpolator
00045 {
00046 public:
00047     unpack_byte_2bit_cpx_samples();
00048     ~unpack_byte_2bit_cpx_samples() = default;
00049     int work(int noutput_items,
00050         gr_vector_const_void_star &input_items,
00051         gr_vector_void_star &output_items);
00052
00053 private:
00054     friend unpack_byte_2bit_cpx_samples_sptr make_unpack_byte_2bit_cpx_samples_sptr();
00055 };
00056
00057
00058 /** \} */
00059 /** \} */
00060 #endif  // GNSS_SDR_UNPACK_BYTE_2BIT_CPX_SAMPLES_H
```

13.386 unpack_byte_2bit_samples.h File Reference

Unpacks byte samples to NSR 2 bits samples.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
```

Classes

- class unpack_byte_2bit_samples

  This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.

Typedefs

- using unpack_byte_2bit_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_samples>

Functions

- unpack_byte_2bit_samples_sptr **make_unpack_byte_2bit_samples** ()

13.386.1 Detailed Description

Unpacks byte samples to NSR 2 bits samples.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file unpack_byte_2bit_samples.h.

13.387 unpack_byte_2bit_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_byte_2bit_samples.h
00003  *
00004  * \brief Unpacks byte samples to NSR 2 bits samples
00005  * \author Javier Arribas jarribas (at) cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_UNPACK_BYTE_2BIT_SAMPLES_H
00019 #define GNSS_SDR_UNPACK_BYTE_2BIT_SAMPLES_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_interpolator.h>
00023
00024
00025 /** \addtogroup Signal_Source
00026  * \{ */
00027 /** \addtogroup Signal_Source_gnuradio_blocks
00028  * \{ */
00029
00030 class unpack_byte_2bit_samples;
00031
00032 using unpack_byte_2bit_samples_sptr = gnss_shared_ptr<unpack_byte_2bit_samples>;
00033
00034 unpack_byte_2bit_samples_sptr make_unpack_byte_2bit_samples();
00035
00036 /*!
00037  * \brief This class implements conversion between byte packet samples to 2bit samples
00038  *  1 byte = 4 2bit samples
00039  */
00040 class unpack_byte_2bit_samples : public gr::sync_interpolator
00041 {
00042 public:
00043     unpack_byte_2bit_samples();
00044     ~unpack_byte_2bit_samples() = default;
00045     int work(int noutput_items,
00046         gr_vector_const_void_star &input_items,
00047         gr_vector_void_star &output_items);
00048
00049 private:
00050     friend unpack_byte_2bit_samples_sptr make_unpack_byte_2bit_samples_sptr();
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_UNPACK_BYTE_2BIT_SAMPLES_H
```

13.388 unpack_byte_4bit_samples.h File Reference

Unpacks byte samples to 4 bits samples. Packing Order Packing order in Nibble I0 I1 I2 I3 I0 I1 I2 I3.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
```

Classes

- class unpack_byte_4bit_samples

  This class implements conversion between byte packet samples to 4bit_cpx samples 1 byte = 1 x complex 4bit I, + 4bit Q samples.

Typedefs

- using unpack_byte_4bit_samples_sptr = gnss_shared_ptr<unpack_byte_4bit_samples>

Functions

- unpack_byte_4bit_samples_sptr **make_unpack_byte_4bit_samples** ()

13.388.1 Detailed Description

Unpacks byte samples to 4 bits samples. Packing Order Packing order in Nibble I0 I1 I2 I3 I0 I1 I2 I3.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file unpack_byte_4bit_samples.h.

13.389 unpack_byte_4bit_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_byte_4bit_samples.h
00003  *
00004  * \brief Unpacks byte samples to 4 bits samples.
00005  *     Packing Order
00006  *     Packing order in Nibble I0 I1 I2 I3 I0 I1 I2 I3
00007  * \author Javier Arribas jarribas (at) cttc.es
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_UNPACK_BYTE_4BIT_SAMPLES_H
00021 #define GNSS_SDR_UNPACK_BYTE_4BIT_SAMPLES_H
00022
00023 #include "gnss_block_interface.h"
00024 #include <gnuradio/sync_interpolator.h>
00025
00026 /** \addtogroup Signal_Source
00027  * \{ */
00028 /** \addtogroup Signal_Source_gnuradio_blocks
00029  * \{ */
00030
00031
00032 class unpack_byte_4bit_samples;
00033
00034 using unpack_byte_4bit_samples_sptr = gnss_shared_ptr<unpack_byte_4bit_samples>;
00035
```

```
00036 unpack_byte_4bit_samples_sptr make_unpack_byte_4bit_samples();
00037
00038 /*!
00039  * \brief This class implements conversion between byte packet samples to 4bit_cpx samples
00040  *  1 byte = 1 x complex 4bit I, + 4bit Q samples
00041  */
00042 class unpack_byte_4bit_samples : public gr::sync_interpolator
00043 {
00044 public:
00045     unpack_byte_4bit_samples();
00046     ~unpack_byte_4bit_samples() = default;
00047     int work(int noutput_items,
00048         gr_vector_const_void_star &input_items,
00049         gr_vector_void_star &output_items);
00050
00051 private:
00052     friend unpack_byte_4bit_samples_sptr make_unpack_byte_4bit_samples_sptr();
00053 };
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_UNPACK_BYTE_4BIT_SAMPLES_H
```

13.390 unpack_gss6450_samples.h File Reference

Unpacks GSS6450 int samples.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
#include <cstdint>
```

Classes

- class unpack_gss6450_samples

Typedefs

- using unpack_gss6450_samples_sptr = gnss_shared_ptr<unpack_gss6450_samples>

Functions

- unpack_gss6450_samples_sptr **make_unpack_gss6450_samples** (int adc_nbit_, bool gss6425_↩ compatible=false)

13.390.1 Detailed Description

Unpacks GSS6450 int samples.

Author

Antonio Ramos, antonio.ramos(at)cttc.es

Javier Arribas jarribas (at) cttc.es

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file unpack_gss6450_samples.h.

13.391 unpack_gss6450_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_gss6450_samples.h
00003  * \brief Unpacks GSS6450 int samples
00004  * \author Antonio Ramos, antonio.ramos(at)cttc.es
00005  * \author Javier Arribas jarribas (at) cttc.es
```

```
00006  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_UNPACK_GSS6450_SAMPLES_H
00020 #define GNSS_SDR_UNPACK_GSS6450_SAMPLES_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/sync_interpolator.h>
00024 #include <cstdint>
00025
00026 /** \addtogroup Signal_Source
00027  * \{ */
00028 /** \addtogroup Signal_Source_gnuradio_blocks
00029  * \{ */
00030
00031
00032 class unpack_gss6450_samples;
00033
00034 using unpack_gss6450_samples_sptr = gnss_shared_ptr<unpack_gss6450_samples>;
00035
00036 unpack_gss6450_samples_sptr make_unpack_gss6450_samples(int adc_nbit_, bool gss6425_compatible =
      false);
00037
00038
00039 class unpack_gss6450_samples : public gr::sync_interpolator
00040 {
00041 public:
00042     explicit unpack_gss6450_samples(int adc_nbit, bool gss6425_compatible = false);
00043     ~unpack_gss6450_samples() = default;
00044     void decode_word(uint32_t input_uint32, gr_complex *out) const;
00045     int work(int noutput_items,
00046         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00047
00048 private:
00049     friend unpack_gss6450_samples_sptr make_unpack_gss6450_samples(int adc_nbit_, bool
      gss6425_compatible);
00050     static int32_t sign_extend(uint32_t value, int bits);
00051     static int32_t decode_gss6425_2bit(uint32_t value);
00052     static int samples_per_gss6450_word(int adc_bits);
00053
00054     int adc_bits_;
00055     int samples_per_int_;
00056     bool gss6425_compatible_;
00057 };
00058
00059
00060 /** \} */
00061 /** \} */
00062 #endif  // GNSS_SDR_UNPACK_GSS6450_SAMPLES_H
```

13.392 unpack_intspir_1bit_samples.h File Reference

Unpacks SPIR int samples to NSR 1 bit samples.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
```

Classes

- class unpack_intspir_1bit_samples

  This class implements conversion between byte packet samples to 2bit samples 1 byte = 4 2bit samples.

Typedefs

- using unpack_intspir_1bit_samples_sptr = gnss_shared_ptr<unpack_intspir_1bit_samples>

Functions

- unpack_intspir_1bit_samples_sptr **make_unpack_intspir_1bit_samples** ()

13.392.1 Detailed Description

Unpacks SPIR int samples to NSR 1 bit samples.

Author

Fran Fabra fabra (at) ice.csic.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is not part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file unpack_intspir_1bit_samples.h.

13.393 unpack_intspir_1bit_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_intspir_1bit_samples.h
00003  *
00004  * \brief Unpacks SPIR int samples to NSR 1 bit samples
00005  * \author Fran Fabra fabra (at) ice.csic.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is not part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_UNPACK_INTSPIR_1BIT_SAMPLES_H
00019 #define GNSS_SDR_UNPACK_INTSPIR_1BIT_SAMPLES_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_interpolator.h>
00023
00024
00025 /** \addtogroup Signal_Source
00026  * \{ */
00027 /** \addtogroup Signal_Source_gnuradio_blocks
00028  * \{ */
00029
00030
00031 class unpack_intspir_1bit_samples;
00032
00033 using unpack_intspir_1bit_samples_sptr = gnss_shared_ptr<unpack_intspir_1bit_samples>;
00034
00035 unpack_intspir_1bit_samples_sptr make_unpack_intspir_1bit_samples();
00036
00037 /*!
00038  * \brief This class implements conversion between byte packet samples to 2bit samples
00039  *  1 byte = 4 2bit samples
00040  */
00041 class unpack_intspir_1bit_samples : public gr::sync_interpolator
00042 {
00043 public:
00044     unpack_intspir_1bit_samples();
00045     ~unpack_intspir_1bit_samples() = default;
00046     int work(int noutput_items,
00047         gr_vector_const_void_star &input_items,
00048         gr_vector_void_star &output_items);
00049
00050 private:
00051     friend unpack_intspir_1bit_samples_sptr make_unpack_intspir_1bit_samples_sptr();
00052 };
00053
00054
00055 /** \} */
00056 /** \} */
00057 #endif  // GNSS_SDR_UNPACK_INTSPIR_1BIT_SAMPLES_H
```

13.394 unpack_ntlab_2bit_samples.h File Reference

Unpacks multichannel 2-bit samples into 4 real-valued floats per input byte.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_interpolator.h>
#include <cstdint>
#include <vector>
```

Classes

- class unpack_ntlab_2bit_samples

  This class implements conversion between byte packet multichannel samples to 2bit samples 1 byte = 4 2bit samples.

Typedefs

- using unpack_ntlab_2bit_samples_sptr = gnss_shared_ptr<unpack_ntlab_2bit_samples>

Functions

- unpack_ntlab_2bit_samples_sptr **make_unpack_ntlab_2bit_samples** (size_t item_size, int nchannels=4)

13.394.1 Detailed Description

Unpacks multichannel 2-bit samples into 4 real-valued floats per input byte.

Author

Pedro Pereira pereirapedrocp (at) gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file unpack_ntlab_2bit_samples.h.

13.395 unpack_ntlab_2bit_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file unpack_ntlab_2bit_samples.h
00003  *
00004  * \brief Unpacks multichannel 2-bit samples into 4 real-valued floats
00005  * per input byte.
00006  * \author Pedro Pereira pereirapedrocp (at) gmail.com
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_UNPACK_NTLAB_2BIT_SAMPLES_H
00020 #define GNSS_SDR_UNPACK_NTLAB_2BIT_SAMPLES_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/sync_interpolator.h>
00024 #include <cstdint>
00025 #include <vector>
00026
00027 /** \addtogroup Signal_Source
00028  * \{ */
00029 /** \addtogroup Signal_Source_gnuradio_blocks
00030  * \{ */
00031
00032
00033 class unpack_ntlab_2bit_samples;
```

```
00034
00035 using unpack_ntlab_2bit_samples_sptr = gnss_shared_ptr<unpack_ntlab_2bit_samples>;
00036
00037 unpack_ntlab_2bit_samples_sptr make_unpack_ntlab_2bit_samples(
00038     size_t item_size,
00039     int nchannels = 4);
00040
00041 /*!
00042  * \brief This class implements conversion between byte packet multichannel samples
00043  *  to 2bit samples 1 byte = 4 2bit samples
00044  *
00045  * Unpack each of the four 2-bit samples in the byte 'b' into four real-valued outputs.
00046  *
00047  * The NTLAB format encodes samples as sign+magnitude pairs in each byte:
00048  * bits 7-6 = [M0 S0] -> sample 0
00049  * bits 5-4 = [M1 S1] -> sample 1
00050  * bits 3-2 = [M2 S2] -> sample 2
00051  * bits 1-0 = [M3 S3] -> sample 3
00052  *
00053  * M = magnitude bit (1->|sample|=3, 0->|sample|=1)
00054  * S = sign     bit (1->positive, 0->negative)
00055  */
00056 class unpack_ntlab_2bit_samples : public gr::sync_interpolator
00057 {
00058 public:
00059     ~unpack_ntlab_2bit_samples() = default;
00060
00061     unpack_ntlab_2bit_samples(size_t item_size,
00062         int nchannels);
00063
00064     int work(int noutput_items,
00065         gr_vector_const_void_star &input_items,
00066         gr_vector_void_star &output_items);
00067
00068 private:
00069     static constexpr int SAMPLES_PER_BYTE = 4;
00070
00071     friend unpack_ntlab_2bit_samples_sptr make_unpack_ntlab_2bit_samples_sptr(
00072         size_t item_size,
00073         int nchannels);
00074
00075     int nchannels_;
00076 };
00077
00078
00079 /** \} */
00080 /** \} */
00081 #endif  // GNSS_SDR_UNPACK_NTLAB_2BIT_SAMPLES_H
```

13.396 unpack_short_byte_samples.h

```
00001 /*!
00002  * \file unpack_short_byte_samples.cc
00003  *
00004  * \brief Unpacks shorts samples to byte samples (1 short = 2 byte samples).
00005  *     Packing Order
00006  *     Packing order in Nibble I0 I1
00007  * \author Javier Arribas jarribas (at) cttc.es
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_UNPACK_SHORT_BYTE_SAMPLES_H
00020 #define GNSS_SDR_UNPACK_SHORT_BYTE_SAMPLES_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/sync_interpolator.h>
00024
00025 /** \addtogroup Signal_Source
00026  * \{ */
00027 /** \addtogroup Signal_Source_gnuradio_blocks
00028  * \{ */
00029
00030
00031 class unpack_short_byte_samples;
00032
00033 using unpack_short_byte_samples_sptr = gnss_shared_ptr<unpack_short_byte_samples>;
```

```
00034
00035 unpack_short_byte_samples_sptr make_unpack_short_byte_samples();
00036
00037 /*!
00038  * \brief This class implements conversion between short packet samples to byte samples
00039  *  1 short = 2 byte samples
00040  */
00041 class unpack_short_byte_samples : public gr::sync_interpolator
00042 {
00043 public:
00044     unpack_short_byte_samples();
00045     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00046     ~unpack_short_byte_samples() = default;
00047     int work(int noutput_items,
00048         gr_vector_const_void_star &input_items,
00049         gr_vector_void_star &output_items);
00050
00051 private:
00052     friend unpack_short_byte_samples_sptr make_unpack_short_byte_samples_sptr();
00053 };
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_UNPACK_SHORT_BYTE_SAMPLES_H
```

13.397 ad9361_manager.h File Reference

An Analog Devices AD9361 front-end configuration library wrapper for configure some functions via iiod link.

```
#include <iio.h>
#include <cstdint>
#include <string>
```

Classes

- struct stream_cfg

Macros

- #define FIR_BUF_SIZE 8192

Enumerations

- enum **iodev** { **RX** , **TX** }

Functions

- void **errchk** (int v, const char *what)
- void **wr_ch_lli** (struct iio_channel *chn, const char *what, int64_t val)
- void **wr_ch_str** (struct iio_channel *chn, const char *what, const char *str)
- struct iio_device * **get_ad9361_phy** (struct iio_context *ctx)
- bool **get_ad9361_stream_dev** (struct iio_context *ctx, enum iodev d, struct iio_device **dev)
- bool **get_ad9361_stream_ch** (struct iio_context *ctx, enum iodev d, struct iio_device *dev, int chid, struct iio_channel **chn)
- bool **get_phy_chan** (struct iio_context *ctx, enum iodev d, int chid, struct iio_channel **chn)
- bool **get_lo_chan** (struct iio_context *ctx, enum iodev d, struct iio_channel **chn)
- bool **cfg_ad9361_streaming_ch** (struct iio_context *ctx, struct stream_cfg *cfg, enum iodev type, int chid)
- bool **config_ad9361_rx_local** (uint64_t bandwidth_, uint64_t sample_rate_, uint64_t freq0_, uint64_t freq1_, const std::string &rf_port_select_, bool rx1_enable_, bool rx2_enable_, const std::string &gain_mode_rx1_, const std::string &gain_mode_rx2_, double rf_gain_rx1_, double rf_gain_rx2_, bool quadrature_, bool rfdc_, bool bbdc_, std::string filter_source_, std::string filter_filename_, float Fpass_, float Fstop_)
- bool **config_ad9361_rx_remote** (const std::string &remote_host, uint64_t bandwidth_, uint64_t sample_↩ rate_, uint64_t freq_, const std::string &rf_port_select_, bool rx1_enable_, bool rx2_enable_, const std::↩ string &gain_mode_rx1_, const std::string &gain_mode_rx2_, double rf_gain_rx1_, double rf_gain_rx2_, bool

quadrature_, bool rfdc_, bool bbdc_, std::string filter_source_, std::string filter_filename_, float Fpass_, float Fstop_)

- bool **config_ad9361_lo_local** (uint64_t bandwidth_, uint64_t sample_rate_, uint64_t freq_rf_tx_hz_, double tx_attenuation_db_, int64_t freq_dds_tx_hz_, double scale_dds_dbfs_, double phase_dds_deg_)
- bool **config_ad9361_lo_remote** (const std::string &remote_host, uint64_t bandwidth_, uint64_t sample_↩ rate_, uint64_t freq_rf_tx_hz_, double tx_attenuation_db_, int64_t freq_dds_tx_hz_, double scale_dds_dbfs_, double phase_dds_deg_)
- bool **ad9361_disable_lo_remote** (const std::string &remote_host)
- bool **ad9361_disable_lo_local** ()
- bool **load_fir_filter** (std::string &filter, struct iio_device ∗phy)
- bool **disable_ad9361_rx_local** ()
- bool **disable_ad9361_rx_remote** (const std::string &remote_host)

13.397.1 Detailed Description

An Analog Devices AD9361 front-end configuration library wrapper for configure some functions via iiod link.

Author

Javier Arribas, jarribas(at)cttc.es

This file contains information taken from librtlsdr: https://git.osmocom.org/rtl-sdr
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ad9361_manager.h.

13.398 ad9361_manager.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ad9361_manager.h
00003  * \brief An Analog Devices AD9361 front-end configuration library wrapper for configure some
      functions via iiod link.
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * This file contains information taken from librtlsdr:
00007  *  https://git.osmocom.org/rtl-sdr
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_AD9361_MANAGER_H
00021 #define GNSS_SDR_AD9361_MANAGER_H
00022
00023 #include <iio.h>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Signal_Source
00028  * \{ */
00029 /** \addtogroup Signal_Source_libs signal_source_libs
00030  * Library with utilities for signal sources.
00031  * \{ */
00032
00033 #define FIR_BUF_SIZE 8192
00034
00035 static const std::string RX_DEV_A = "ad9361-phy";          // one or two AD9361s are present
00036 static const std::string RX_DEV_B = "ad9361-phy-B";        // one or two AD9361s are present
00037 static const std::string RX_STREAM_DEV = "cf-ad9361-lpc";  // one AD9361 is present
00038 static const std::string RX_STREAM_DEV_A = "cf-ad9361-A";  // two AD9361s are present
00039 static const std::string RX_STREAM_DEV_B = "cf-ad9361-B";  // two AD9361s are present
00040 static const std::string TX_STREAM_DEV = "cf-ad9361-dds-core-lpc";
00041
00042 /* RX is input, TX is output */
00043 enum iodev
```

```
00044 {
00045     RX,
00046     TX
00047 };
00048
00049 /* common RX and TX streaming params */
00050 struct stream_cfg
00051 {
00052     int64_t bw_hz;       // Analog bandwidth in Hz
00053     int64_t fs_hz;       // Baseband sample rate in Hz
00054     int64_t lo_hz;       // Local oscillator frequency in Hz
00055     const char *rfport;  // Port name
00056 };
00057
00058 /* check return value of attr_write function */
00059 void errchk(int v, const char *what);
00060
00061 /* write attribute: int64_t int */
00062 void wr_ch_lli(struct iio_channel *chn, const char *what, int64_t val);
00063
00064 /* write attribute: string */
00065 void wr_ch_str(struct iio_channel *chn, const char *what, const char *str);
00066
00067 /* returns ad9361 phy device */
00068 struct iio_device *get_ad9361_phy(struct iio_context *ctx);
00069
00070 /* finds AD9361 streaming IIO devices */
00071 bool get_ad9361_stream_dev(struct iio_context *ctx, enum iodev d, struct iio_device **dev);
00072
00073 /* finds AD9361 streaming IIO channels */
00074 bool get_ad9361_stream_ch(struct iio_context *ctx, enum iodev d, struct iio_device *dev, int chid,
      struct iio_channel **chn);
00075
00076 /* finds AD9361 phy IIO configuration channel with id chid */
00077 bool get_phy_chan(struct iio_context *ctx, enum iodev d, int chid, struct iio_channel **chn);
00078
00079 /* finds AD9361 local oscillator IIO configuration channels */
00080 bool get_lo_chan(struct iio_context *ctx, enum iodev d, struct iio_channel **chn);
00081
00082 /* applies streaming configuration through IIO */
00083 bool cfg_ad9361_streaming_ch(struct iio_context *ctx, struct stream_cfg *cfg, enum iodev type, int
      chid);
00084
00085 bool config_ad9361_rx_local(uint64_t bandwidth_,
00086     uint64_t sample_rate_,
00087     uint64_t freq0_,
00088     uint64_t freq1_,
00089     const std::string &rf_port_select_,
00090     bool rx1_enable_,
00091     bool rx2_enable_,
00092     const std::string &gain_mode_rx1_,
00093     const std::string &gain_mode_rx2_,
00094     double rf_gain_rx1_,
00095     double rf_gain_rx2_,
00096     bool quadrature_,
00097     bool rfdc_,
00098     bool bbdc_,
00099     std::string filter_source_,
00100     std::string filter_filename_,
00101     float Fpass_,
00102     float Fstop_);
00103
00104 bool config_ad9361_rx_remote(const std::string &remote_host,
00105     uint64_t bandwidth_,
00106     uint64_t sample_rate_,
00107     uint64_t freq_,
00108     const std::string &rf_port_select_,
00109     bool rx1_enable_,
00110     bool rx2_enable_,
00111     const std::string &gain_mode_rx1_,
00112     const std::string &gain_mode_rx2_,
00113     double rf_gain_rx1_,
00114     double rf_gain_rx2_,
00115     bool quadrature_,
00116     bool rfdc_,
00117     bool bbdc_,
00118     std::string filter_source_,
00119     std::string filter_filename_,
00120     float Fpass_,
00121     float Fstop_);
00122
00123 bool config_ad9361_lo_local(uint64_t bandwidth_,
00124     uint64_t sample_rate_,
00125     uint64_t freq_rf_tx_hz_,
00126     double tx_attenuation_db_,
00127     int64_t freq_dds_tx_hz_,
00128     double scale_dds_dbfs_,
```

```
00129     double phase_dds_deg_);
00130
00131 bool config_ad9361_lo_remote(const std::string &remote_host,
00132     uint64_t bandwidth_,
00133     uint64_t sample_rate_,
00134     uint64_t freq_rf_tx_hz_,
00135     double tx_attenuation_db_,
00136     int64_t freq_dds_tx_hz_,
00137     double scale_dds_dbfs_,
00138     double phase_dds_deg_);
00139
00140 bool ad9361_disable_lo_remote(const std::string &remote_host);
00141
00142 bool ad9361_disable_lo_local();
00143
00144 bool load_fir_filter(std::string &filter, struct iio_device *phy);
00145
00146 bool disable_ad9361_rx_local();
00147
00148 bool disable_ad9361_rx_remote(const std::string &remote_host);
00149
00150
00151 /** \} */
00152 /** \} */
00153 #endif  // GNSS_SDR_AD9361_MANAGER_H
```

13.399 ad936x_iio_custom.h File Reference

A direct IIO custom front-end driver for the AD936x AD front-end family with special FPGA custom functionalities.

```
#include "ad936x_iio_samples.h"
#include "concurrent_queue.h"
#include "gnss_time.h"
#include "pps_samplestamp.h"
#include <boost/atomic.hpp>
#include <iio.h>
#include <ad9361.h>
#include <memory>
#include <string>
#include <thread>
#include <vector>
```

Classes

- class ad936x_iio_custom

13.399.1 Detailed Description

A direct IIO custom front-end driver for the AD936x AD front-end family with special FPGA custom functionalities.

13.399.1.1 autotoc_md397

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ad936x_iio_custom.h.

13.400 ad936x_iio_custom.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ad936x_iio_custom.h
00003  * \brief A direct IIO custom front-end driver for the AD936x AD front-end
00004  * family with special FPGA custom functionalities.
00005  * \author Javier Arribas, jarribas(at)cttc.es
```

```
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_AD936X_IIO_CUSTOM_H
00019 #define GNSS_SDR_AD936X_IIO_CUSTOM_H
00020
00021 #include "ad936x_iio_samples.h"
00022 #include "concurrent_queue.h"
00023 #include "gnss_time.h"
00024 #include "pps_samplestamp.h"
00025 #include <boost/atomic.hpp>
00026 #include <iio.h>
00027 #include <ad9361.h>  // multichip sync and high level functions
00028 #include <memory>
00029 #include <string>
00030 #include <thread>
00031 #include <vector>
00032
00033 /** \addtogroup Signal_Source
00034  * \{ */
00035 /** \addtogroup Signal_Source_libs
00036  * \{ */
00037
00038
00039 class ad936x_iio_custom
00040 {
00041 public:
00042     ad936x_iio_custom(int debug_level_, int log_level_);
00043     virtual ~ad936x_iio_custom();
00044     bool initialize_device(const std::string &pluto_device_uri, const std::string &board_type);
00045
00046     bool init_config_ad9361_rx(long long bandwidth_,
00047         long long sample_rate_,
00048         long long freq_,
00049         const std::string &rf_port_select_,
00050         const std::string &rf_filter,
00051         const std::string &gain_mode_rx0_,
00052         const std::string &gain_mode_rx1_,
00053         double rf_gain_rx0_,
00054         double rf_gain_rx1_,
00055         bool enable_ch0,
00056         bool enable_ch1,
00057         long long freq_2ch,
00058         double lo_attenuation_db_,
00059         bool high_side_lo_,
00060         int tx_lo_channel_);
00061
00062     bool calibrate(int ch, double bw_hz);
00063
00064     double get_rx_gain(int ch_num);
00065     bool setRXGain(int ch_num, const std::string &gain_mode, double gain_dB);
00066
00067     bool set_antenna_port(int ch, int antenna_idx);
00068     double get_frequency(int ch);
00069     bool set_frequency(int ch, double freq_hz);
00070
00071     bool start_sample_rx(bool ppsmode);
00072     void stop_record();
00073
00074     void set_gnsstime_queue(std::shared_ptr<Concurrent_Queue<GnssTime>> queue);
00075     void set_pps_samplestamp_queue(std::shared_ptr<Concurrent_Queue<PpsSamplestamp>> queue);
00076
00077     bool get_rx_frequency(long long &freq_hz);
00078     bool set_rx_frequency(long long freq_hz);
00079     bool read_die_temp(double &temp_c);
00080
00081     void pop_sample_buffer(std::shared_ptr<ad936x_iio_samples> &current_buffer);
00082
00083     void push_sample_buffer(std::shared_ptr<ad936x_iio_samples> &current_buffer);
00084     int n_channels;
00085
00086 private:
00087     std::shared_ptr<Concurrent_Queue<GnssTime» GnssTime_queue;
00088     std::shared_ptr<Concurrent_Queue<PpsSamplestamp» Pps_queue;
00089     bool check_device();
00090     bool get_iio_param(iio_device *dev, const std::string &param, std::string &value);
00091     void configure_params(struct iio_device *phy,
00092         const std::vector<std::string> &params);
```

```
00093     void set_params_rx(struct iio_device *phy_device,
00094         unsigned long long frequency,
00095         unsigned long samplerate, unsigned long bandwidth,
00096         bool quadrature, bool rfdc, bool bbdc,
00097         const std::string &gain1, double gain1_value,
00098         const std::string &gain2, double gain2_value,
00099         const std::string &port_select);
00100
00101     bool config_ad9361_dds(uint64_t freq_rf_tx_hz_,
00102         double tx_attenuation_db_,
00103         int64_t freq_dds_tx_hz_,
00104         double scale_dds_,
00105         double phase_dds_deg_,
00106         int channel);
00107
00108     void get_PPS_timestamp();
00109     void capture(const std::vector<std::string> &channels);
00110
00111     bool select_rf_filter(const std::string &rf_filter);
00112
00113     void monitor_thread_fn();
00114
00115     void PlutoTxEnable(bool txon);
00116     void setPlutoGpo(int p);
00117
00118     // Device structure
00119     struct iio_context *ctx;
00120     struct iio_device *phy;
00121     struct iio_device *stream_dev;
00122     struct iio_device *dds_dev;
00123
00124     std::mutex mtx;
00125     std::condition_variable cv;
00126
00127     boost::atomic<bool> receive_samples;
00128
00129     boost::atomic<bool> fpga_overflow;
00130     // using queues of smart pointers to preallocated buffers
00131     Concurrent_Queue<std::shared_ptr<ad936x_iio_samples>> free_buffers;
00132     Concurrent_Queue<std::shared_ptr<ad936x_iio_samples>> used_buffers;
00133
00134     std::thread capture_samples_thread;
00135     std::thread overflow_monitor_thread;
00136     std::thread capture_time_thread;
00137
00138     // stream
00139     uint64_t sample_rate_sps;
00140     int debug_level;
00141     int log_level;
00142     bool PPS_mode;
00143 };
00144
00145 /** \} */
00146 /** \} */
00147 #endif  // GNSS_SDR_AD936X_IIO_CUSTOM_H
```

13.401 ad936x_iio_samples.h File Reference

A class that holds a custom sample buffer for Analog Devices AD936x family front-ends.

```
#include <cstdint>
#include <memory>
#include <vector>
```

Classes

- class ad936x_iio_samples

Macros

- #define IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES 32768 * 4
- #define IIO_INPUTRAMFIFOSIZE 256

13.401.1 Detailed Description

A class that holds a custom sample buffer for Analog Devices AD936x family front-ends.

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ad936x_iio_samples.h.

13.401.2 Macro Definition Documentation

13.401.2.1 IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES

```
#define IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES 32768 * 4
```

Definition at line 21 of file ad936x_iio_samples.h.

13.401.2.2 IIO_INPUTRAMFIFOSIZE

```
#define IIO_INPUTRAMFIFOSIZE 256
```

Definition at line 22 of file ad936x_iio_samples.h.

13.402 ad936x_iio_samples.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ad936x_iio_samples.h
00003  * \brief A class that holds a custom sample buffer for Analog Devices AD936x family front-ends.
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_AD936X_IIO_SAMPLES_H
00019 #define GNSS_SDR_AD936X_IIO_SAMPLES_H
00020
00021 #define IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES 32768 * 4
00022 #define IIO_INPUTRAMFIFOSIZE 256
00023
00024 #include <cstdint>
00025 #include <memory>
00026 #include <vector>
00027
00028 /** \addtogroup Signal_Source
00029  * \{ */
00030 /** \addtogroup Signal_Source_libs
00031  * \{ */
00032
00033 class ad936x_iio_samples
00034 {
00035 public:
00036     ad936x_iio_samples() = default;
00037     uint32_t n_bytes{0};
00038     uint32_t n_interleaved_iq_samples{0};
00039     uint16_t n_channels{0};
00040     uint16_t step_bytes{0};
00041     char buffer[IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES * 4 * 4];  // max 16 bits samples per buffer (4
      channels, 2-bytes per I + 2-bytes per Q)
00042 };
00043
00044 /** \} */
00045 /** \} */
00046 #endif
```

13.403 fpga_buffer_monitor.h File Reference

Check receiver buffer overflow and monitor the status of the receiver buffers.

```
#include <cstdint>
#include <fstream>
#include <string>
```

Classes

- class Fpga_buffer_monitor

  Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers.

13.403.1 Detailed Description

Check receiver buffer overflow and monitor the status of the receiver buffers.

Authors

- Marc Majoral, 2021. mmajoral(at)cttc.es

Class that checks the receiver buffer overflow flags and monitors the status of the receiver buffers.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file fpga_buffer_monitor.h.

13.404 fpga_buffer_monitor.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_buffer_monitor.h
00003  * \brief Check receiver buffer overflow and monitor the status of the receiver
00004  * buffers.
00005  * \authors
00006  * <ul>
00007  *    <li> Marc Majoral, 2021. mmajoral(at)cttc.es
00008  * </ul>
00009  *
00010  * Class that checks the receiver buffer overflow flags and monitors the status
00011  * of the receiver buffers.
00012  *
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00017  * This file is part of GNSS-SDR.
00018  *
00019  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00020  * SPDX-License-Identifier: GPL-3.0-or-later
00021  *
00022  * -----------------------------------------------------------------------------
00023  */
00024
00025 #ifndef GNSS_SDR_FPGA_BUFFER_MONITOR_H
00026 #define GNSS_SDR_FPGA_BUFFER_MONITOR_H
00027
00028 #include <cstdint>  // for int32_t
00029 #include <fstream>  // for std::ofstream
00030 #include <string>   // for std::string
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_libs
00035  * \{ */
00036
00037
00038 /*!
00039  * \brief Class that checks the receiver buffer overflow flags and monitors the
00040  * status of the receiver buffers.
00041  */
00042 class Fpga_buffer_monitor
00043 {
00044 public:
00045     /*!
00046      * \brief Constructor
00047      */
00048     explicit Fpga_buffer_monitor(uint32_t num_freq_bands,
00049         bool dump,
00050         std::string dump_filename);
```

```
00051     //    explicit Fpga_buffer_monitor(const std::string& device_name,
00052     //        uint32_t num_freq_bands,
00053     //        bool dump,
00054     //        std::string dump_filename);
00055
00056     /*!
00057      * \brief Destructor
00058      */
00059     ~Fpga_buffer_monitor();
00060
00061     /*!
00062      * \brief This function checks buffer overflow and monitors the FPGA buffer status
00063      */
00064     bool check_buffer_overflow_and_monitor_buffer_status();
00065
00066 private:
00067     const std::string BUFFER_MONITOR_DEVICE_NAME = std::string("buffer_monitor");  // buffer monitor
      device name
00068     static const size_t FPGA_PAGE_SIZE = 0x1000;
00069     static const uint32_t test_register_writeval = 0x55AA;
00070     static const uint32_t num_sapmples_per_buffer_element = 2;
00071     // write addresses
00072     static const uint32_t reset_overflow_flags_and_max_buff_size_reg_addr = 0;
00073     // read-write addresses
00074     static const uint32_t test_reg_addr = 7;
00075     // read addresses
00076     static const uint32_t current_buff_occ_freq_band_0_reg_addr = 0;
00077     static const uint32_t current_buff_occ_freq_band_1_reg_addr = 1;
00078     static const uint32_t max_buff_occ_freq_band_0_reg_addr = 2;
00079     static const uint32_t max_buff_occ_freq_band_1_reg_addr = 3;
00080     static const uint32_t overflow_flags_reg_addr = 4;
00081     // FPGA-related constants
00082     static const uint32_t overflow_freq_band_0_bit_pos = 1;
00083     static const uint32_t overflow_freq_band_1_bit_pos = 2;
00084
00085     int32_t buffer_monitor_test_register();
00086     void close_device();
00087
00088     std::string d_dump_filename;
00089     std::ofstream d_dump_file;
00090
00091     volatile unsigned* d_map_base;  // driver memory map corresponding to the FPGA buffer monitor
00092     int d_device_descriptor;        // driver descriptor corresponding to the FPGA buffer monitor
00093
00094     uint32_t d_num_freq_bands;
00095
00096     uint32_t d_max_buff_occ_freq_band_0;
00097     uint32_t d_max_buff_occ_freq_band_1;
00098
00099     bool d_dump;
00100 };
00101
00102
00103 /** \} */
00104 /** \} */
00105 #endif  // GNSS_SDR_FPGA_BUFFER_MONITOR_H
```

13.405 fpga_dma-proxy.h File Reference

FPGA DMA control. This code is based in the Xilinx DMA proxy test application: https://github.com/Xilinx-Wiki-Projects/software-prototypes/tree/master/linux-user-space-dma/Software.

```
#include <cstdint>
```

Classes

- class Fpga_DMA

  Class that controls DMA transfers to the FPGA.

Enumerations

- enum **proxy_status** { **PROXY_NO_ERROR** = 0 , **PROXY_BUSY** = 1 , **PROXY_TIMEOUT** = 2 , **PROXY_ERROR** = 3 }

Variables

- int8_t buffer [DMA_MAX_BUFFER_SIZE]
- enum proxy_status **status**
- unsigned int length

13.405.1 Detailed Description

FPGA DMA control. This code is based in the Xilinx DMA proxy test application: https://github.com/Xilinx-Wiki-Projects/software-prototypes/tree/master/linux-user-space-dma/Software.

Author

Marc Majoral, mmajoral(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fpga_dma-proxy.h.

13.405.2 Enumeration Type Documentation

13.405.2.1 proxy_status

```
enum proxy_status
```
Definition at line 1 of file fpga_dma-proxy.h.

13.405.3 Variable Documentation

13.405.3.1 buffer

```
int8_t buffer[DMA_MAX_BUFFER_SIZE]
```
Definition at line 0 of file fpga_dma-proxy.h.

13.405.3.2 length

```
unsigned int length
```
Definition at line 8 of file fpga_dma-proxy.h.

13.406 fpga_dma-proxy.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_dma-proxy.h
00003  * \brief FPGA DMA control. This code is based in the Xilinx DMA proxy test application:
00004  *
      https://github.com/Xilinx-Wiki-Projects/software-prototypes/tree/master/linux-user-space-dma/Software
00005  * \author Marc Majoral, mmajoral(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_FPGA_DMA_PROXY_H
00019 #define GNSS_SDR_FPGA_DMA_PROXY_H
00020
00021 #include <cstdint>  // for std::int8_t
00022
00023 /*!
00024  * \brief Class that controls DMA transfers to the FPGA
00025  */
00026 class Fpga_DMA
00027 {
```

```
00028 public:
00029     /*!
00030      * \brief Default constructor.
00031      */
00032     Fpga_DMA() = default;
00033
00034     /*!
00035      * \brief Release any open DMA resources.
00036      */
00037     ~Fpga_DMA();
00038
00039     Fpga_DMA(const Fpga_DMA &) = delete;
00040     Fpga_DMA &operator=(const Fpga_DMA &) = delete;
00041
00042     /*!
00043      * \brief Open the DMA device driver.
00044      */
00045     int DMA_open(void);
00046
00047     /*!
00048      * \brief Obtain DMA buffer address.
00049      */
00050     int8_t *get_buffer_address() const;
00051
00052     /*!
00053      * \brief Obtain DMA buffer size.
00054      */
00055     uint32_t get_buffer_size() const;
00056
00057     /*!
00058      * \brief Transfer DMA data
00059      */
00060     int DMA_write(int nbytes) const;
00061
00062     /*!
00063      * \brief Close the DMA device driver
00064      */
00065     int DMA_close(void);
00066
00067 private:
00068     static const uint32_t DMA_MAX_BUFFER_SIZE = (128 * 1024); /* must match driver exactly */
00069     static const uint32_t TX_BUFFER_COUNT = 1;
00070
00071     // channel buffer structure
00072     struct channel_buffer
00073     {
00074         int8_t buffer[DMA_MAX_BUFFER_SIZE];
00075         enum proxy_status
00076         {
00077             PROXY_NO_ERROR = 0,
00078             PROXY_BUSY = 1,
00079             PROXY_TIMEOUT = 2,
00080             PROXY_ERROR = 3
00081         } status;
00082         unsigned int length;
00083     } __attribute__((aligned(1024))); /* 64 byte alignment required for DMA, but 1024 handy for
    viewing memory */
00084
00085     // internal DMA channel data structure
00086     struct channel
00087     {
00088         struct channel_buffer *buf_ptr;
00089         int fd;
00090     };
00091
00092     channel tx_channel{nullptr, -1};
00093 };
00094 #endif  // GNSS_SDR_FPGA_DMA_PROXY_H
```

13.407 fpga_dynamic_bit_selection.h File Reference

Dynamic bit selection in the received signal.

```
#include <cstddef>
#include <cstdint>
#include <string>
```

Classes

- class Fpga_dynamic_bit_selection

Controls dynamic bit selection in the FPGA.

13.407.1 Detailed Description

Dynamic bit selection in the received signal.

Authors

- Marc Majoral, 2020. mmajoral(at)cttc.es

Class that controls the Dynamic Bit Selection in the FPGA.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fpga_dynamic_bit_selection.h.

13.408 fpga_dynamic_bit_selection.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_dynamic_bit_selection.h
00003  * \brief Dynamic bit selection in the received signal.
00004  * \authors <ul>
00005  *          <li> Marc Majoral, 2020. mmajoral(at)cttc.es
00006  *          </ul>
00007  *
00008  * Class that controls the Dynamic Bit Selection in the FPGA.
00009  *
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_FPGA_DYNAMIC_BIT_SELECTION_H
00023 #define GNSS_SDR_FPGA_DYNAMIC_BIT_SELECTION_H
00024
00025 #include <cstddef>
00026 #include <cstdint>
00027 #include <string>
00028
00029 /** \addtogroup Signal_Source
00030  * \{ */
00031 /** \addtogroup Signal_Source_libs
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief Controls dynamic bit selection in the FPGA.
00037  */
00038 class Fpga_dynamic_bit_selection
00039 {
00040 public:
00041     /*!
00042      * \brief Constructor
00043      */
00044     explicit Fpga_dynamic_bit_selection(bool enable_rx1_band, bool enable_rx2_band);
00045
00046     /*!
00047      * \brief Destructor
00048      */
00049     ~Fpga_dynamic_bit_selection();
00050
00051     // Prevent copying objects that own FPGA mappings and file descriptors.
00052     Fpga_dynamic_bit_selection(const Fpga_dynamic_bit_selection &) = delete;
00053     Fpga_dynamic_bit_selection &operator=(const Fpga_dynamic_bit_selection &) = delete;
00054
00055
00056     /*!
00057      * \brief Adjusts the bit shift for each enabled frequency band based on signal power.
00058      */
00059     void bit_selection(void);
00060
00061 private:
00062     // IP core device name
```

```
00063     const std::string DYN_BIT_SEL_DEV_NAME = std::string("dynamic_bits_selector");  // Dynamic bit
      selection device name
00064
00065     // IP Core version
00066     const uint32_t FPGA_DYN_BIT_SEL_IP_VERSION_1_2 = 0x0002;  // Dynamic bit selection IP core version
      1.2
00067
00068     // page size
00069     static const size_t FPGA_PAGE_SIZE = 0x1000;
00070
00071     // write-only registers
00072     static const uint32_t SOBITS_REG_ADDR = 0;  // number of shift bits
00073
00074     // read-only registers
00075     static const uint32_t FPGA_IP_CORE_VERSION_REG_ADDR = 0;  // IP core version register address
00076     static const uint32_t SIGPOW_REG_ADDR = 1;                // rx signal power
00077     static const uint32_t MAX_BIT_SHIFT_REG_ADDR = 2;         // maximum number of shift bits
00078
00079     // dynamic bit selection parameters
00080     static const uint32_t SELECT_FREQ_BAND_1 = 1;          // selection for frequency band 1
00081     static const uint32_t SELECT_FREQ_BAND_2 = 2;          // selection for frequency band 2
00082     static const uint32_t SHIFT_OUT_BITS_MAX_DEFAULT = 8;  // take the most significant bits by
      default
00083     static const uint32_t SHIFT_OUT_BITS_MIN = 0;          // minimum possible value for the bit
      selection
00084     static const uint32_t POWER_THRESHOLD_HIGH = 9000;
00085     static const uint32_t POWER_THRESHOLD_LOW = 3000;
00086
00087     bool open_device(volatile unsigned **d_map_base, int &d_dev_descr, int freq_band);
00088     void initialize_device(volatile unsigned *d_map_base, uint32_t &shift_out_bits, uint32_t
      &shift_out_bit_max);
00089     void bit_selection_per_rf_band(volatile unsigned *d_map_base, uint32_t &shift_out_bits, uint32_t
      shift_out_bit_max);
00090     void close_device(volatile unsigned *d_map_base, int &d_dev_descr);
00091
00092     volatile unsigned *d_map_base_freq_band_1;
00093     volatile unsigned *d_map_base_freq_band_2;
00094     int d_dev_descr_freq_band_1;
00095     int d_dev_descr_freq_band_2;
00096     uint32_t d_shift_out_bits_freq_band_1;
00097     uint32_t d_shift_out_bits_freq_band_2;
00098     uint32_t d_shift_out_bit_max_band_1;
00099     uint32_t d_shift_out_bit_max_band_2;
00100     bool d_enable_rx1_band;
00101     bool d_enable_rx2_band;
00102 };
00103
00104
00105 /** \} */
00106 /** \} */
00107 #endif  // GNSS_SDR_FPGA_DYNAMIC_BIT_SELECTION_H
```

13.409 fpga_spidev.h File Reference

FPGA SPI control.

```
#include <cstdint>
#include <string>
```

Classes

- class Fpga_spidev

13.409.1 Detailed Description

FPGA SPI control.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file fpga_spidev.h.

13.410 fpga_spidev.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_spidev.h
```

```
00003  * \brief FPGA SPI control.
00004  *
00005  * -----------------------------------------------------------------------------
00006  *
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00011  * SPDX-License-Identifier: GPL-3.0-or-later
00012  *
00013  * -----------------------------------------------------------------------------
00014  */
00015
00016 #ifndef GNSS_SDR_FPGA_SPIDEV_H
00017 #define GNSS_SDR_FPGA_SPIDEV_H
00018
00019 #include <cstdint>
00020 #include <string>
00021
00022 class Fpga_spidev
00023 {
00024 public:
00025     /*!
00026      * \brief Default constructor.
00027      */
00028     Fpga_spidev() = default;
00029
00030     /*!
00031      * \brief Default destructor.
00032      */
00033     ~Fpga_spidev() = default;
00034
00035     /*!
00036      * \brief write a register through the SPI.
00037      */
00038     int write_reg32(char addr, uint32_t data);
00039
00040     /*!
00041      * \brief read a register through the SPI.
00042      */
00043     int read_reg32(uint8_t addr, uint32_t* copy_to);
00044     /*!
00045      * \brief Open the SPI device driver.
00046      */
00047     int SPI_open(std::string spi_device_name);
00048
00049     /*!
00050      * \brief Close the SPI device driver
00051      */
00052     int SPI_close(void) const;
00053
00054 private:
00055     static const uint32_t SPI_SPEED = 250000;
00056     // const std::string SPI_DEVICE_NAME = std::string("/dev/spidev2.0");  // Switch UIO device name
00057
00058     int d_fd;
00059 };
00060
00061
00062 #endif  // GNSS_SDR_FPGA_SPIDEV_H
```

13.411 fpga_switch.h File Reference

Switch that connects the HW accelerator queues to the analog front end or the DMA.

```
#include <cstdint>
#include <string>
```

Classes

- class Fpga_Switch

  Class that controls the switch in the FPGA, which connects the FPGA acquisition and multicorrelator modules to either the DMA or the Analog Front-End.

13.411.1 Detailed Description

Switch that connects the HW accelerator queues to the analog front end or the DMA.

Authors

- Marc Majoral, 2019. mmajoral(at)cttc.cat
- Javier Arribas, 2016. jarribas(at)cttc.es

Class that controls a switch in the FPGA
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fpga_switch.h.

13.412 fpga_switch.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_switch.h
00003  * \brief Switch that connects the HW accelerator queues to the analog front end or the DMA.
00004  * \authors <ul>
00005  *          <li> Marc Majoral, 2019. mmajoral(at)cttc.cat
00006  *          <li> Javier Arribas, 2016. jarribas(at)cttc.es
00007  *          </ul>
00008  *
00009  * Class that controls a switch in the FPGA
00010  *
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_FPGA_SWITCH_H
00024 #define GNSS_SDR_FPGA_SWITCH_H
00025
00026 #include <cstdint>
00027 #include <string>
00028
00029 /** \addtogroup Signal_Source
00030  * \{ */
00031 /** \addtogroup Signal_Source_libs
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief Class that controls the switch in the FPGA, which connects the FPGA acquisition and
      multicorrelator modules to
00037  * either the DMA or the Analog Front-End.
00038  */
00039 class Fpga_Switch
00040 {
00041 public:
00042     /*!
00043      * \brief Constructor
00044      */
00045     Fpga_Switch(void);
00046     /*!
00047      * \brief Destructor
00048      */
00049     ~Fpga_Switch();
00050
00051     /*!
00052      * \brief This function configures the switch in th eFPGA
00053      */
00054     void set_switch_position(int32_t switch_position);
00055
00056 private:
00057     const std::string SWITCH_DEVICE_NAME = std::string("AXIS_Switch_v1_0_0");  // Switch UIO device
      name
00058     static const size_t FPGA_PAGE_SIZE = 0x1000;
00059     static const uint32_t TEST_REGISTER_TRACK_WRITEVAL = 0x55AA;
00060     static const uint32_t MAX_LENGTH_DEVICEIO_NAME = 50;
00061
00062     // private functions
00063     unsigned fpga_switch_test_register(unsigned writeval);
00064     void close_device(void);
00065
00066     volatile unsigned* d_map_base;  // driver memory map
00067     int d_device_descriptor;        // driver descriptor
```

```
00068 };
00069
00070
00071 /** \} */
00072 /** \} */
00073 #endif  // GNSS_SDR_FPGA_SWITCH_H
```

13.413 gnss_sdr_timestamp.h File Reference

GNURadio block that adds to sample stream timestamp metadata information stored on a sepparated file.

```
#include "gnss_block_interface.h"
#include "gnss_time.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <string>
```

Classes

- class Gnss_Sdr_Timestamp

Functions

- gnss_shared_ptr< Gnss_Sdr_Timestamp > **gnss_sdr_make_Timestamp** (size_t sizeof_stream_item, std::string timestamp_file, double clock_offset_ms, int items_to_samples)

13.413.1 Detailed Description

GNURadio block that adds to sample stream timestamp metadata information stored on a sepparated file.

Author

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_sdr_timestamp.h.

13.414 gnss_sdr_timestamp.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_timestamp.h
00003  * \brief  GNURadio block that adds to sample stream timestamp metadata information stored on a
      sepparated file
00004  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00005  *
00006  * -------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_TIMESTAMP_H
00019 #define GNSS_SDR_GNSS_SDR_TIMESTAMP_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "gnss_time.h"
```

```
00023 #include <gnuradio/sync_block.h>  // for sync_block
00024 #include <gnuradio/types.h>       // for gr_vector_const_void_star
00025 #include <pmt/pmt.h>
00026 #include <cstddef>  // for size_t
00027 #include <cstdint>
00028 #include <fstream>
00029 #include <string>
00030
00031 /** \addtogroup Signal_Source
00032  * \{ */
00033 /** \addtogroup Signal_Source_libs
00034  * \{ */
00035
00036
00037 class Gnss_Sdr_Timestamp;
00038
00039 gnss_shared_ptr<Gnss_Sdr_Timestamp> gnss_sdr_make_Timestamp(
00040     size_t sizeof_stream_item,
00041     std::string timestamp_file,
00042     double clock_offset_ms,
00043     int items_to_samples);
00044
00045
00046 class Gnss_Sdr_Timestamp : public gr::sync_block
00047 {
00048 public:
00049     int work(int noutput_items,
00050         gr_vector_const_void_star& input_items,
00051         gr_vector_void_star& output_items);
00052     bool start();
00053
00054 private:
00055     friend gnss_shared_ptr<Gnss_Sdr_Timestamp> gnss_sdr_make_Timestamp(
00056         size_t sizeof_stream_item,
00057         std::string timestamp_file,
00058         double clock_offset_ms,
00059         int items_to_samples);
00060
00061     Gnss_Sdr_Timestamp(size_t sizeof_stream_item,
00062         std::string timestamp_file,
00063         double clock_offset_ms,
00064         int items_to_samples);
00065
00066     int64_t uint64diff(uint64_t first, uint64_t second);
00067     bool read_next_timetag();
00068     std::string d_timefile;
00069     std::fstream d_timefilestream;
00070     GnssTime next_timetag{};
00071     double d_clock_offset_ms;
00072     double d_fraction_ms_offset;
00073     double d_integer_ms_offset;
00074     int d_items_to_samples;
00075     uint64_t d_next_timetag_samplecount;
00076     bool d_get_next_timetag;
00077 };
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_GNSS_SDR_TIMESTAMP_H
```

13.415 gnss_sdr_valve.h File Reference

Interface of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.

```
#include "concurrent_queue.h"
#include "gnss_block_interface.h"
#include <gnuradio/sync_block.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <cstddef>
#include <cstdint>
```

Classes

- class Gnss_Sdr_Valve

Implementation of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.

Functions

- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue, bool stop_flowgraph)
- gnss_shared_ptr< Gnss_Sdr_Valve > **gnss_sdr_make_valve** (size_t sizeof_stream_item, uint64_t nitems, Concurrent_Queue< pmt::pmt_t > *queue, bool stop_flowgraph, int n_streams)

13.415.1 Detailed Description

Interface of a GNU Radio block that sends a STOP message to the control queue right after a specific number of samples have passed through it.

Author

Javier Arribas, 2018. jarribas(at)cttc.es

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_sdr_valve.h.

13.416 gnss_sdr_valve.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_valve.h
00003  * \brief  Interface of a GNU Radio block that sends a STOP message to the
00004  * control queue right after a specific number of samples have passed through it.
00005  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00006  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00007  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020
00021 #ifndef GNSS_SDR_GNSS_SDR_VALVE_H
00022 #define GNSS_SDR_GNSS_SDR_VALVE_H
00023
00024 #include "concurrent_queue.h"
00025 #include "gnss_block_interface.h"
00026 #include <gnuradio/sync_block.h>  // for sync_block
00027 #include <gnuradio/types.h>       // for gr_vector_const_void_star
00028 #include <pmt/pmt.h>
00029 #include <cstddef>  // for size_t
00030 #include <cstdint>
00031
00032 /** \addtogroup Signal_Source
00033  * \{ */
00034 /** \addtogroup Signal_Source_libs
00035  * \{ */
00036
00037
00038 class Gnss_Sdr_Valve;
00039
00040 gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00041     size_t sizeof_stream_item,
00042     uint64_t nitems,
00043     Concurrent_Queue<pmt::pmt_t>* queue);
```

```
00044
00045 gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00046     size_t sizeof_stream_item,
00047     uint64_t nitems,
00048     Concurrent_Queue<pmt::pmt_t>* queue,
00049     bool stop_flowgraph);
00050
00051 gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00052     size_t sizeof_stream_item,
00053     uint64_t nitems,
00054     Concurrent_Queue<pmt::pmt_t>* queue,
00055     bool stop_flowgraph,
00056     int n_streams);
00057
00058 /*!
00059  * \brief Implementation of a GNU Radio block that sends a STOP message to the
00060  * control queue right after a specific number of samples have passed through it.
00061  */
00062 class Gnss_Sdr_Valve : public gr::sync_block
00063 {
00064 public:
00065     void open_valve();
00066
00067     int work(int noutput_items,
00068         gr_vector_const_void_star& input_items,
00069         gr_vector_void_star& output_items);
00070
00071 private:
00072     friend gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00073         size_t sizeof_stream_item,
00074         uint64_t nitems,
00075         Concurrent_Queue<pmt::pmt_t>* queue);
00076
00077     friend gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00078         size_t sizeof_stream_item,
00079         uint64_t nitems,
00080         Concurrent_Queue<pmt::pmt_t>* queue,
00081         bool stop_flowgraph);
00082
00083     friend gnss_shared_ptr<Gnss_Sdr_Valve> gnss_sdr_make_valve(
00084         size_t sizeof_stream_item,
00085         uint64_t nitems,
00086         Concurrent_Queue<pmt::pmt_t>* queue,
00087         bool stop_flowgraph,
00088         int n_streams);
00089
00090     Gnss_Sdr_Valve(size_t sizeof_stream_item,
00091         uint64_t nitems,
00092         Concurrent_Queue<pmt::pmt_t>* queue,
00093         bool stop_flowgraph,
00094         int n_streams);
00095
00096     uint64_t d_nitems;
00097     uint64_t d_ncopied_items;
00098     Concurrent_Queue<pmt::pmt_t>* d_queue;
00099     bool d_stop_flowgraph;
00100     bool d_open_valve;
00101 };
00102
00103
00104 /** \} */
00105 /** \} */
00106 #endif  // GNSS_SDR_GNSS_SDR_VALVE_H
```

13.417 ion_gsms_chunk_data.h File Reference

Holds logic for reading and decoding samples from a chunk.

```
#include "ion_gsms_chunk_unpacking_ctx.h"
#include "ion_gsms_stream_encodings.h"
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <GnssMetadata.h>
#include <cstddef>
#include <cstdint>
#include <iostream>
#include <stdexcept>
#include <string>
#include <vector>
```

Classes

- struct Allocator
- struct Deleter
- class IONGSMSChunkData

Functions

- std::size_t bits_to_item_size (std::size_t bit_count)
- template<typename Callback>
  void with_word_type (std::size_t word_size, Callback callback)

13.417.1 Detailed Description

Holds logic for reading and decoding samples from a chunk.

Author

Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com

Carles Fernandez, 2026 carles.fernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ion_gsms_chunk_data.h.

13.417.2 Function Documentation

13.417.2.1 bits_to_item_size()

```
std::size_t bits_to_item_size (
            std::size_t bit_count)  [inline]
```

Definition at line 34 of file ion_gsms_chunk_data.h.

13.417.2.2 with_word_type()

```
template<typename Callback>
void with_word_type (
            std::size_t word_size,
            Callback callback)
```

Definition at line 91 of file ion_gsms_chunk_data.h.

13.418 ion_gsms_chunk_data.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ion_gsms_chunk_data.h
00003  * \brief Holds logic for reading and decoding samples from a chunk
00004  * \author Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com
00005  * \author Carles Fernandez, 2026 carles.fernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_ION_GSMS_CHUNK_DATA_H
00019 #define GNSS_SDR_ION_GSMS_CHUNK_DATA_H
00020
```

```
00021 #include "ion_gsms_chunk_unpacking_ctx.h"
00022 #include "ion_gsms_stream_encodings.h"
00023 #include <gnuradio/block.h>
00024 #include <gnuradio/gr_complex.h>
00025 #include <GnssMetadata.h>
00026 #include <cstddef>
00027 #include <cstdint>
00028 #include <iostream>
00029 #include <stdexcept>
00030 #include <string>
00031 #include <vector>
00032
00033
00034 inline std::size_t bits_to_item_size(std::size_t bit_count)
00035 {
00036     if (bit_count <= 8)
00037         {
00038             return 1;
00039         }
00040     if (bit_count <= 16)
00041         {
00042             return 2;
00043         }
00044     if (bit_count <= 32)
00045         {
00046             return 4;
00047         }
00048     if (bit_count <= 64)
00049         {
00050             return 8;
00051         }
00052
00053     throw std::runtime_error("ION_GSMS_Signal_Source item size is larger than 64 bits");
00054 }
00055
00056
00057 // Define a functor that has a templated operator()
00058 struct Allocator
00059 {
00060     std::size_t countwords_;
00061     void*& buffer_;  // Using void* to hold any type of pointer
00062
00063     Allocator(std::size_t countwords, void*& buffer)
00064         : countwords_(countwords), buffer_(buffer) {}
00065
00066     template <typename WordType>
00067     void operator()() const
00068     {
00069         buffer_ = new WordType[countwords_];
00070     }
00071 };
00072
00073
00074 // Define a functor to delete the allocated memory
00075 struct Deleter
00076 {
00077     void* buffer_;
00078
00079     explicit Deleter(void* buffer)
00080         : buffer_(buffer) {}
00081
00082     template <typename WordType>
00083     void operator()() const
00084     {
00085         delete[] static_cast<WordType*>(buffer_);
00086     }
00087 };
00088
00089
00090 template <typename Callback>
00091 void with_word_type(std::size_t word_size, Callback callback)
00092 {
00093     switch (word_size)
00094         {
00095         case 1:
00096             callback.template operator()<uint8_t>();
00097             break;
00098         case 2:
00099             callback.template operator()<uint16_t>();
00100             break;
00101         case 4:
00102             callback.template operator()<uint32_t>();
00103             break;
00104         case 8:
00105             callback.template operator()<uint64_t>();
00106             break;
00107         default:
```

```
00108             throw std::runtime_error("ION_GSMS_Signal_Source unsupported chunk word size: " +
      std::to_string(word_size));
00109         }
00110 }
00111
00112 class IONGSMSChunkData
00113 {
00114 public:
00115     IONGSMSChunkData(const GnssMetadata::Chunk& chunk, const std::vector<std::string>& stream_ids,
      std::size_t output_stream_offset);
00116
00117     ~IONGSMSChunkData();
00118
00119     IONGSMSChunkData(const IONGSMSChunkData& rhl) = delete;
00120     IONGSMSChunkData& operator=(const IONGSMSChunkData& rhl) = delete;
00121
00122     IONGSMSChunkData(IONGSMSChunkData&& rhl) = delete;
00123     IONGSMSChunkData& operator=(IONGSMSChunkData&& rhl) = delete;
00124
00125     std::size_t read_from_buffer(uint8_t* buffer, std::size_t offset);
00126
00127     void write_to_output(gr_vector_void_star& outputs, std::vector<int>& output_items);
00128
00129     std::size_t output_stream_count() const;
00130     std::size_t output_stream_item_size(std::size_t stream_index) const;
00131     std::size_t output_stream_item_rate(std::size_t stream_index) const;
00132
00133     static bool stream_is_complex(GnssMetadata::IonStream::SampleFormat format);
00134     static uint8_t stream_output_item_bits(const GnssMetadata::IonStream& stream);
00135     static std::size_t stream_output_item_size(const GnssMetadata::IonStream& stream);
00136     static std::size_t stream_output_item_rate(const GnssMetadata::IonStream& stream);
00137
00138 private:
00139     template <typename WT>
00140     void unpack_words(gr_vector_void_star& outputs, std::vector<int>& output_items);
00141
00142     template <typename WT>
00143     std::size_t write_stream_samples(
00144         IONGSMSChunkUnpackingCtx<WT>& ctx,
00145         const GnssMetadata::Lump& lump,
00146         const GnssMetadata::IonStream& stream,
00147         GnssMetadata::StreamEncoding stream_encoding,
00148         void** out);
00149
00150     template <typename WT, typename OT>
00151     std::size_t write_n_samples(
00152         IONGSMSChunkUnpackingCtx<WT>& ctx,
00153         GnssMetadata::Lump::LumpShift lump_shift,
00154         const GnssMetadata::IonStream& stream,
00155         GnssMetadata::StreamEncoding stream_encoding,
00156         OT** out);
00157
00158     template <typename WT>
00159     std::size_t write_fp32_samples(
00160         IONGSMSChunkUnpackingCtx<WT>& ctx,
00161         GnssMetadata::Lump::LumpShift lump_shift,
00162         const GnssMetadata::IonStream& stream,
00163         void** out);
00164
00165     template <typename Sample>
00166     static Sample decode_sample(uint8_t sample_bitsize, uint64_t raw_sample,
      GnssMetadata::StreamEncoding encoding);
00167
00168     template <typename WT>
00169     static float read_fp32_sample(IONGSMSChunkUnpackingCtx<WT>& ctx);
00170
00171     static gr_complex complex_sample_from_format(GnssMetadata::IonStream::SampleFormat format, float
      first, float second);
00172     static bool stream_outputs_gr_complex(const GnssMetadata::IonStream& stream);
00173     static bool samples_are_reversed(GnssMetadata::IonStream::StreamShift stream_shift,
      GnssMetadata::Lump::LumpShift lump_shift);
00174     static std::size_t stream_padding_bits(const GnssMetadata::IonStream& stream);
00175     static uint64_t low_bits_mask(std::size_t bits);
00176
00177     template <typename Word>
00178     static Word byte_swap_word(Word value);
00179
00180     static bool host_is_little_endian();
00181     static bool source_endianness_is_different(GnssMetadata::Chunk::WordEndian endian);
00182
00183     const GnssMetadata::Chunk& chunk_;
00184     std::size_t sizeword_;
00185     std::size_t countwords_;
00186     std::size_t padding_bitsize_;
00187     std::size_t output_stream_offset_;
00188     std::size_t output_stream_count_;
00189     std::vector<std::size_t> output_stream_item_size_;
```

```
00190     std::vector<std::size_t> output_stream_item_rate_;
00191
00192     struct stream_metadata_t
00193     {
00194         const GnssMetadata::Lump& lump;
00195         const GnssMetadata::IonStream& stream;
00196         GnssMetadata::StreamEncoding stream_encoding;
00197         int output_index = -1;
00198         std::size_t output_item_offset = 0;
00199         std::size_t output_item_size = 0;
00200
00201         stream_metadata_t(
00202             const GnssMetadata::Lump& lump_,
00203             const GnssMetadata::IonStream& stream_,
00204             GnssMetadata::StreamEncoding stream_encoding_,
00205             int output_index_ = -1,
00206             std::size_t output_item_offset_ = 0,
00207             std::size_t output_item_size_ = 0) : lump(lump_),
00208                                                  stream(stream_),
00209                                                  stream_encoding(stream_encoding_),
00210                                                  output_index(output_index_),
00211                                                  output_item_offset(output_item_offset_),
00212                                                  output_item_size(output_item_size_)
00213         {
00214         }
00215     };
00216     std::vector<stream_metadata_t> streams_;
00217     std::vector<int> output_stream_indices_;
00218
00219     void* buffer_ = nullptr;
00220 };
00221
00222 #endif  // GNSS_SDR_ION_GSMS_CHUNK_DATA_H
```

13.419 ion_gsms_chunk_unpacking_ctx.h File Reference

Holds state and provides utilities for unpacking samples from a chunk.

```
#include <cstddef>
#include <cstdint>
#include <stdexcept>
```

Classes

- struct IONGSMSChunkUnpackingCtx< WT >

13.419.1 Detailed Description

Holds state and provides utilities for unpacking samples from a chunk.

Author

Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com

This is a template class, and thus, its member functions must be defined in the header file.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ion_gsms_chunk_unpacking_ctx.h.

13.420 ion_gsms_chunk_unpacking_ctx.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ion_gsms_chunk_unpacking_ctx.h
00003  * \brief Holds state and provides utilities for unpacking samples from a chunk
00004  * \author Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * This is a template class, and thus, its member functions must be defined in the header file.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
```

```
00012  *
00013  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_ION_GSMS_CHUNK_UNPACKING_CTX_H
00020 #define GNSS_SDR_ION_GSMS_CHUNK_UNPACKING_CTX_H
00021
00022 #include <cstddef>
00023 #include <cstdint>
00024 #include <stdexcept>
00025
00026 /** \addtogroup Signal_Source
00027  * \{ */
00028 /** \addtogroup Signal_Source_libs
00029  * \{ */
00030
00031 template <typename WT>
00032 struct IONGSMSChunkUnpackingCtx
00033 {
00034     static constexpr uint8_t word_bitsize_ = sizeof(WT) * 8;
00035
00036     const WT* data_ = nullptr;  // Not owned by this class, MUST NOT destroy.
00037     std::size_t word_count_ = 0;
00038     std::size_t bit_offset_ = 0;
00039     bool read_lsb_first_ = false;
00040
00041     IONGSMSChunkUnpackingCtx(
00042         WT* data_buffer,
00043         std::size_t data_buffer_word_count,
00044         bool read_lsb_first) : data_(data_buffer),
00045                                word_count_(data_buffer_word_count),
00046                                read_lsb_first_(read_lsb_first)
00047     {
00048     }
00049
00050     void shift_padding(std::size_t n_bits)
00051     {
00052         bit_offset_ += n_bits;
00053     }
00054
00055     uint64_t read_bits(std::size_t bit_count)
00056     {
00057         uint64_t value = 0;
00058         for (std::size_t bit = 0; bit < bit_count; ++bit)
00059             {
00060                 const std::size_t absolute_bit = bit_offset_ + bit;
00061                 const std::size_t word_index = absolute_bit / word_bitsize_;
00062                 if (word_index >= word_count_)
00063                     {
00064                         throw std::runtime_error("ION_GSMS_Signal_Source tried to read past the chunk
      boundary");
00065                     }
00066                 const std::size_t bit_index = absolute_bit % word_bitsize_;
00067                 const std::size_t word_bit = read_lsb_first_ ? bit_index : word_bitsize_ - 1 -
      bit_index;
00068                 const uint64_t bit_value = (static_cast<uint64_t>(data_[word_index]) » word_bit) &
      0x01U;
00069                 if (read_lsb_first_)
00070                     {
00071                         value |= bit_value « bit;
00072                     }
00073                 else
00074                     {
00075                         value «= 1;
00076                         value |= bit_value;
00077                     }
00078             }
00079
00080         bit_offset_ += bit_count;
00081         return value;
00082     }
00083 };
00084
00085 /** \} */
00086 /** \} */
00087 #endif  // GNSS_SDR_ION_GSMS_CHUNK_UNPACKING_CTX_H
```

13.421 ion_gsms_stream_encodings.h File Reference

Implements look up tables for all encodings in the standard.

```
#include <string>
```

Typedefs

- using GnssMetadata::StreamEncoding = unsigned char

Functions

- StreamEncoding GnssMetadata::encoding_from_string (const std::string &str)

Variables

- constexpr unsigned char GnssMetadata::StreamEncodings::SIGN = 0
- constexpr unsigned char GnssMetadata::StreamEncodings::OB = 1
- constexpr unsigned char GnssMetadata::StreamEncodings::SM = 2
- constexpr unsigned char GnssMetadata::StreamEncodings::MS = 3
- constexpr unsigned char GnssMetadata::StreamEncodings::TC = 4
- constexpr unsigned char GnssMetadata::StreamEncodings::OG = 5
- constexpr unsigned char GnssMetadata::StreamEncodings::OBA = 6
- constexpr unsigned char GnssMetadata::StreamEncodings::SMA = 7
- constexpr unsigned char GnssMetadata::StreamEncodings::MSA = 8
- constexpr unsigned char GnssMetadata::StreamEncodings::TCA = 9
- constexpr unsigned char GnssMetadata::StreamEncodings::OGA = 10
- constexpr unsigned char GnssMetadata::StreamEncodings::FP = 11
- constexpr unsigned char GnssMetadata::StreamEncodings::UNKNOWN = 255
- template<typename T>
  T GnssMetadata::two_bit_look_up [11][4]
- template<typename T>
  T GnssMetadata::three_bit_look_up [11][8]
- template<typename T>
  T GnssMetadata::four_bit_look_up [11][16]
- template<typename T>
  T GnssMetadata::five_bit_look_up [11][32]

13.421.1 Detailed Description

Implements look up tables for all encodings in the standard.

Author

Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com

These tables are taken from the stardard's official document.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file ion_gsms_stream_encodings.h.

13.421.2 Typedef Documentation

13.421.2.1 StreamEncoding

```
using GnssMetadata::StreamEncoding = unsigned char
```

Definition at line 32 of file ion_gsms_stream_encodings.h.

13.421.3 Function Documentation

13.421.3.1 encoding_from_string()

```
StreamEncoding GnssMetadata::encoding_from_string (
            const std::string & str)  [inline]
```

Definition at line 53 of file ion_gsms_stream_encodings.h.

13.421.4 Variable Documentation

13.421.4.1 five_bit_look_up

```
template<typename T>
T GnssMetadata::five_bit_look_up[11][32]  [inline]
```

Initial value:

```
{
    {},

    {-16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
      10, 11, 12, 13, 14, 15},
    {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0, -1, -2, -3, -4, -5, -6, -7, -8, -9, -10, -11,
      -12, -13, -14, -15},
    {0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1,
      -1},
    {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, -16, -15, -14, -13, -12, -11, -10, -9, -8, -7,
      -6, -5, -4, -3, -2, -1},
    {-16, -15, -13, -14, -9, -10, -12, -11, -1, -2, -4, -3, -8, -7, -5, -6, 15, 14, 12, 13, 8, 9, 11, 10, 0,
      1, 3, 2, 7, 6, 4, 5},
    {-31, -29, -27, -25, -23, -21, -19, -17, -15, -13, -11, -9, -7, -5, -3, -1, 1, 3, 5, 7, 9, 11, 13, 15,
      17, 19, 21, 23, 25, 27, 29, 31},
    {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, -1, -3, -5, -7, -9, -11, -13, -15, -17, -19,
      -21, -23, -25, -27, -29, -31},
    {1, -1, 3, -3, 5, -5, 7, -7, 9, -9, 11, -11, 13, -13, 15, -15, 17, -17, 19, -19, 21, -21, 23, -23, 25,
      -25, 27, -27, 29, -29, 31, -31},
    {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, -31, -29, -27, -25, -23, -21, -19, -17, -15,
      -13, -11, -9, -7, -5, -3, -1},
    {-31, -29, -25, -27, -17, -19, -23, -21, -1, -3, -7, -5, -15, -13, -9, -11, 31, 29, 25, 27, 17, 19, 23,
      21, 1, 3, 7, 5, 15, 13, 9, 11},
}
```

Definition at line 152 of file ion_gsms_stream_encodings.h.

13.421.4.2 four_bit_look_up

```
template<typename T>
T GnssMetadata::four_bit_look_up[11][16]  [inline]
```

Initial value:

```
{
    {},
    {-8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7},
    {0, 1, 2, 3, 4, 5, 6, 7, 0, -1, -2, -3, -4, -5, -6, -7},
    {0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1},
    {0, 1, 2, 3, 4, 5, 6, 7, -8, -7, -6, -5, -4, -3, -2, -1},
    {-8, -7, -5, -6, -1, -2, -4, -3, 7, 6, 4, 5, 0, 1, 3, 2},
    {-15, -13, -11, -9, -7, -5, -3, -1, 1, 3, 5, 7, 9, 11, 13, 15},
    {1, 3, 5, 7, 9, 11, 13, 15, -1, -3, -5, -7, -9, -11, -13, -15},
    {1, -1, 3, -3, 5, -5, 7, -7, 9, -9, 11, -11, 13, -13, 15, -15},
    {1, 3, 5, 7, 9, 11, 13, 15, -15, -13, -11, -9, -7, -5, -3, -1},
    {-15, -13, -9, -11, -1, -3, -7, -5, 15, 13, 9, 11, 1, 3, 7, 5},
}
```

Definition at line 137 of file ion_gsms_stream_encodings.h.

13.421.4.3 FP

```
unsigned char GnssMetadata::StreamEncodings::FP = 11  [constexpr]
```

Definition at line 48 of file ion_gsms_stream_encodings.h.

13.421.4.4 MS

```
unsigned char GnssMetadata::StreamEncodings::MS = 3  [constexpr]
```

Definition at line 40 of file ion_gsms_stream_encodings.h.

13.421.4.5 MSA

```
unsigned char GnssMetadata::StreamEncodings::MSA = 8  [constexpr]
```

Definition at line 45 of file ion_gsms_stream_encodings.h.

13.421.4.6 OB

```
unsigned char GnssMetadata::StreamEncodings::OB = 1  [constexpr]
```

Definition at line 38 of file ion_gsms_stream_encodings.h.

13.421.4.7 OBA

```
unsigned char GnssMetadata::StreamEncodings::OBA = 6  [constexpr]
```
Definition at line 43 of file ion_gsms_stream_encodings.h.

13.421.4.8 OG

```
unsigned char GnssMetadata::StreamEncodings::OG = 5  [constexpr]
```
Definition at line 42 of file ion_gsms_stream_encodings.h.

13.421.4.9 OGA

```
unsigned char GnssMetadata::StreamEncodings::OGA = 10  [constexpr]
```
Definition at line 47 of file ion_gsms_stream_encodings.h.

13.421.4.10 SIGN

```
unsigned char GnssMetadata::StreamEncodings::SIGN = 0  [constexpr]
```
Definition at line 37 of file ion_gsms_stream_encodings.h.

13.421.4.11 SM

```
unsigned char GnssMetadata::StreamEncodings::SM = 2  [constexpr]
```
Definition at line 39 of file ion_gsms_stream_encodings.h.

13.421.4.12 SMA

```
unsigned char GnssMetadata::StreamEncodings::SMA = 7  [constexpr]
```
Definition at line 44 of file ion_gsms_stream_encodings.h.

13.421.4.13 TC

```
unsigned char GnssMetadata::StreamEncodings::TC = 4  [constexpr]
```
Definition at line 41 of file ion_gsms_stream_encodings.h.

13.421.4.14 TCA

```
unsigned char GnssMetadata::StreamEncodings::TCA = 9  [constexpr]
```
Definition at line 46 of file ion_gsms_stream_encodings.h.

13.421.4.15 three_bit_look_up

```
template<typename T>
T GnssMetadata::three_bit_look_up[11][8]  [inline]
```
Initial value:
```
{
    {},
    {-4, -3, -2, -1, 0, 1, 2, 3},
    {0, 1, 2, 3, 0, -1, -2, -3},
    {0, 0, 1, -1, 0, 0, 1, -1},
    {0, 1, 2, 3, -4, -3, -2, -1},
    {-4, -3, -1, -2, 3, 2, 0, 1},
    {-7, -5, -3, -1, 1, 3, 5, 7},
    {1, 3, 5, 7, -1, -3, -5, -7},
    {1, -1, 3, -3, 5, -5, 7, -7},
    {1, 3, 5, 7, -7, -5, -3, -1},
    {-7, -5, -1, -3, 7, 5, 1, 3},
}
```
Definition at line 122 of file ion_gsms_stream_encodings.h.

13.421.4.16 two_bit_look_up

```
template<typename T>
T GnssMetadata::two_bit_look_up[11][4]  [inline]
```

Initial value:

```
{
    {},
    {-2, -1, 0, 1},
    {0, 1, 0, -1},
    {0, 0, 1, -1},
    {0, 1, -2, -1},
    {-2, -1, 1, 0},
    {-3, -1, 1, 3},
    {1, 3, -1, -3},
    {1, -1, 3, -3},
    {1, 3, -3, -1},
    {-3, -1, 3, 1},
}
```

Definition at line 107 of file ion_gsms_stream_encodings.h.

13.421.4.17 UNKNOWN

```
unsigned char GnssMetadata::StreamEncodings::UNKNOWN = 255  [constexpr]
```

Definition at line 49 of file ion_gsms_stream_encodings.h.

13.422 ion_gsms_stream_encodings.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ion_gsms_stream_encodings.h
00003  * \brief Implements look up tables for all encodings in the standard
00004  * \author Víctor Castillo Agüero, 2024. victorcastilloaguero(at)gmail.com
00005  *
00006  * These tables are taken from the stardard's official document.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_ION_GSMS_STREAM_ENCODINGS_H
00020 #define GNSS_SDR_ION_GSMS_STREAM_ENCODINGS_H
00021
00022 #include <string>
00023
00024 /** \addtogroup Signal_Source
00025  * \{ */
00026 /** \addtogroup Signal_Source_libs
00027  * \{ */
00028
00029 namespace GnssMetadata
00030 {
00031
00032 using StreamEncoding = unsigned char;
00033
00034 namespace StreamEncodings
00035 {
00036
00037 constexpr unsigned char SIGN = 0;
00038 constexpr unsigned char OB = 1;
00039 constexpr unsigned char SM = 2;
00040 constexpr unsigned char MS = 3;
00041 constexpr unsigned char TC = 4;
00042 constexpr unsigned char OG = 5;
00043 constexpr unsigned char OBA = 6;
00044 constexpr unsigned char SMA = 7;
00045 constexpr unsigned char MSA = 8;
00046 constexpr unsigned char TCA = 9;
00047 constexpr unsigned char OGA = 10;
00048 constexpr unsigned char FP = 11;
00049 constexpr unsigned char UNKNOWN = 255;
00050
00051 }  // namespace StreamEncodings
```

```
00052
00053 inline StreamEncoding encoding_from_string(const std::string& str)
00054 {
00055     if (str == "SIGN")
00056         {
00057             return StreamEncodings::SIGN;
00058         }
00059     if (str == "OB")
00060         {
00061             return StreamEncodings::OB;
00062         }
00063     if (str == "SM")
00064         {
00065             return StreamEncodings::SM;
00066         }
00067     if (str == "MS")
00068         {
00069             return StreamEncodings::MS;
00070         }
00071     if (str == "TC")
00072         {
00073             return StreamEncodings::TC;
00074         }
00075     if (str == "OG")
00076         {
00077             return StreamEncodings::OG;
00078         }
00079     if (str == "OBA")
00080         {
00081             return StreamEncodings::OBA;
00082         }
00083     if (str == "SMA")
00084         {
00085             return StreamEncodings::SMA;
00086         }
00087     if (str == "MSA")
00088         {
00089             return StreamEncodings::MSA;
00090         }
00091     if (str == "TCA")
00092         {
00093             return StreamEncodings::TCA;
00094         }
00095     if (str == "OGA")
00096         {
00097             return StreamEncodings::OGA;
00098         }
00099     if (str == "FP")
00100         {
00101             return StreamEncodings::FP;
00102         }
00103     return StreamEncodings::UNKNOWN;
00104 }
00105
00106 template <typename T>
00107 inline T two_bit_look_up[11][4]{
00108     {},                 // [0]
00109     {-2, -1, 0, 1},   // [1   /*OB*/]
00110     {0, 1, 0, -1},    // [2   /*SM*/]
00111     {0, 0, 1, -1},    // [3   /*MS*/]
00112     {0, 1, -2, -1},   // [4   /*TC*/]
00113     {-2, -1, 1, 0},   // [5   /*OG*/]
00114     {-3, -1, 1, 3},   // [6  /*OBA*/]
00115     {1, 3, -1, -3},   // [7  /*SMA*/]
00116     {1, -1, 3, -3},   // [8  /*MSA*/]
00117     {1, 3, -3, -1},   // [9  /*TCA*/]
00118     {-3, -1, 3, 1},   // [10 /*OGA*/]
00119 };
00120
00121 template <typename T>
00122 inline T three_bit_look_up[11][8]{
00123     {},                                 // [0]
00124     {-4, -3, -2, -1, 0, 1, 2, 3},   // [1   /*OB*/]
00125     {0, 1, 2, 3, 0, -1, -2, -3},    // [2   /*SM*/]
00126     {0, 0, 1, -1, 0, 0, 1, -1},     // [3   /*MS*/]
00127     {0, 1, 2, 3, -4, -3, -2, -1},   // [4   /*TC*/]
00128     {-4, -3, -1, -2, 3, 2, 0, 1},   // [5   /*OG*/]
00129     {-7, -5, -3, -1, 1, 3, 5, 7},   // [6  /*OBA*/]
00130     {1, 3, 5, 7, -1, -3, -5, -7},   // [7  /*SMA*/]
00131     {1, -1, 3, -3, 5, -5, 7, -7},   // [8  /*MSA*/]
00132     {1, 3, 5, 7, -7, -5, -3, -1},   // [9  /*TCA*/]
00133     {-7, -5, -1, -3, 7, 5, 1, 3},   // [10 /*OGA*/]
00134 };
00135
00136 template <typename T>
00137 inline T four_bit_look_up[11][16]{
00138     {},                                                                 // [0]
```

```
00139     {-8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7},         // [1   /*OB*/]
00140     {0, 1, 2, 3, 4, 5, 6, 7, 0, -1, -2, -3, -4, -5, -6, -7},          // [2   /*SM*/]
00141     {0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1},             // [3   /*MS*/]
00142     {0, 1, 2, 3, 4, 5, 6, 7, -8, -7, -6, -5, -4, -3, -2, -1},         // [4   /*TC*/]
00143     {-8, -7, -5, -6, -1, -2, -4, -3, 7, 6, 4, 5, 0, 1, 3, 2},         // [5   /*OG*/]
00144     {-15, -13, -11, -9, -7, -5, -3, -1, 1, 3, 5, 7, 9, 11, 13, 15},   // [6  /*OBA*/]
00145     {1, 3, 5, 7, 9, 11, 13, 15, -1, -3, -5, -7, -9, -11, -13, -15},   // [7  /*SMA*/]
00146     {1, -1, 3, -3, 5, -5, 7, -7, 9, -9, 11, -11, 13, -13, 15, -15},   // [8  /*MSA*/]
00147     {1, 3, 5, 7, 9, 11, 13, 15, -15, -13, -11, -9, -7, -5, -3, -1},   // [9  /*TCA*/]
00148     {-15, -13, -9, -11, -1, -3, -7, -5, 15, 13, 9, 11, 1, 3, 7, 5},   // [10 /*OGA*/]
00149 };
00150
00151 template <typename T>
00152 inline T five_bit_look_up[11][32]{
00153     {},
      // [0]
00154     {-16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8,
      9, 10, 11, 12, 13, 14, 15},            // [1   /*OB*/]
00155     {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0, -1, -2, -3, -4, -5, -6, -7, -8, -9, -10,
      -11, -12, -13, -14, -15},              // [2   /*SM*/]
00156     {0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0,
      1, -1},                                // [3   /*MS*/]
00157     {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, -16, -15, -14, -13, -12, -11, -10, -9, -8,
      -7, -6, -5, -4, -3, -2, -1},           // [4   /*TC*/]
00158     {-16, -15, -13, -14, -9, -10, -12, -11, -1, -2, -4, -3, -8, -7, -5, -6, 15, 14, 12, 13, 8, 9, 11,
      10, 0, 1, 3, 2, 7, 6, 4, 5},           // [5   /*OG*/]
00159     {-31, -29, -27, -25, -23, -21, -19, -17, -15, -13, -11, -9, -7, -5, -3, -1, 1, 3, 5, 7, 9, 11, 13,
      15, 17, 19, 21, 23, 25, 27, 29, 31},   // [6  /*OBA*/]
00160     {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, -1, -3, -5, -7, -9, -11, -13, -15,
      -17, -19, -21, -23, -25, -27, -29, -31},  // [7  /*SMA*/]
00161     {1, -1, 3, -3, 5, -5, 7, -7, 9, -9, 11, -11, 13, -13, 15, -15, 17, -17, 19, -19, 21, -21, 23, -23,
      25, -25, 27, -27, 29, -29, 31, -31},   // [8  /*MSA*/]
00162     {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, -31, -29, -27, -25, -23, -21, -19,
      -17, -15, -13, -11, -9, -7, -5, -3, -1},  // [9  /*TCA*/]
00163     {-31, -29, -25, -27, -17, -19, -23, -21, -1, -3, -7, -5, -15, -13, -9, -11, 31, 29, 25, 27, 17,
      19, 23, 21, 1, 3, 7, 5, 15, 13, 9, 11},  // [10 /*OGA*/]
00164 };
00165
00166 }  // namespace GnssMetadata
00167
00168
00169 /** \} */
00170 /** \} */
00171 #endif  // GNSS_SDR_ION_GSMS_STREAM_ENCODINGS_H
```

13.423 pps_samplestamp.h File Reference

A simple container for the sample count associated to PPS rising edge.

`#include <cstdint>`

Classes

- class PpsSamplestamp

13.423.1 Detailed Description

A simple container for the sample count associated to PPS rising edge.

13.423.1.1 autotoc_md421

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file pps_samplestamp.h.

13.424 pps_samplestamp.h

Go to the documentation of this file.

```
00001 /*!
```

```
00002  * \file pps_samplestamp.h
00003  * \brief A simple container for the sample count associated to PPS rising edge
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  * -----------------------------------------------------------------------------
00006  *
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00011  * SPDX-License-Identifier: GPL-3.0-or-later
00012  *
00013  * -----------------------------------------------------------------------------
00014  */
00015
00016 #ifndef IIOPPS_PPS_SAMPLESTAMP_H
00017 #define IIOPPS_PPS_SAMPLESTAMP_H
00018
00019 #include <cstdint>
00020
00021 class PpsSamplestamp
00022 {
00023 public:
00024     uint64_t samplestamp = 0;   // PPS rising edge samples counter from the beginning of rx stream
      operation. Notice that it is reset to zero if sample buffer overflow is detected on the FPGA side
00025     uint32_t overflow_reg = 0;  // >0 indicates overflow situation in the FPGA RX buffer
00026 };
00027
00028 #endif
```

13.425 ppstcprx.h File Reference

TCP client class for front-end PPS samplestamp information reception.

```
#include "concurrent_queue.h"
#include "pps_samplestamp.h"
#include <arpa/inet.h>
#include <memory>
#include <netinet/in.h>
#include <string>
#include <sys/socket.h>
#include <sys/types.h>
```

Classes

- class pps_tcp_rx

13.425.1 Detailed Description

TCP client class for front-end PPS samplestamp information reception.

13.425.1.1 autotoc_md423

Author

Javier Arribas, jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file ppstcprx.h.

13.426 ppstcprx.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ppstcprx.h
00003  * \brief TCP client class for front-end PPS samplestamp information reception
00004  * \author Javier Arribas, jarribas(at)cttc.es
00005  * -----------------------------------------------------------------------------
00006  *
```

```
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00011  * SPDX-License-Identifier: GPL-3.0-or-later
00012  *
00013  * ------------------------------------------------------------------------
00014  */
00015
00016 #ifndef SRC_LIBS_PPSTCPRX_H_
00017 #define SRC_LIBS_PPSTCPRX_H_
00018 #include "concurrent_queue.h"
00019 #include "pps_samplestamp.h"
00020 #include <arpa/inet.h>
00021 #include <memory>
00022 #include <netinet/in.h>
00023 #include <string>
00024 #include <sys/socket.h>
00025 #include <sys/types.h>
00026
00027 class pps_tcp_rx
00028 {
00029 private:
00030     std::shared_ptr<Concurrent_Queue<PpsSamplestamp» Pps_queue;
00031     int clientSd{-1};
00032
00033 public:
00034     volatile bool is_connected{false};
00035     pps_tcp_rx() = default;
00036     virtual ~pps_tcp_rx() = default;
00037
00038     void receive_pps(const std::string& ip_address, int port);
00039     bool send_cmd(std::string cmd) const;
00040
00041     void set_pps_samplestamp_queue(std::shared_ptr<Concurrent_Queue<PpsSamplestamp>> queue);
00042 };
00043
00044 #endif /* SRC_LIBS_PPSTCPRX_H_ */
```

13.427 rtl_tcp_commands.h File Reference

Defines structures and constants for communicating with rtl_tcp.

```
#include <boost/asio/ip/tcp.hpp>
#include <boost/system/error_code.hpp>
```

Enumerations

- enum RTL_TCP_COMMAND {
  RTL_TCP_SET_FREQUENCY = 1 , **RTL_TCP_SET_SAMPLE_RATE** = 2 , **RTL_TCP_SET_GAIN_MODE** = 3 , **RTL_TCP_SET_GAIN** = 4 ,
  RTL_TCP_SET_IF_GAIN = 6 , **RTL_TCP_SET_AGC_MODE** = 8 }

  Command IDs for configuration rtl_tcp.

Functions

- boost::system::error_code rtl_tcp_command (RTL_TCP_COMMAND id, unsigned param, boost::asio::ip::↩ tcp::socket &socket)

  Send a command to rtl_tcp over the given socket.

13.427.1 Detailed Description

Defines structures and constants for communicating with rtl_tcp.

Author

Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au

This file contains information taken from librtlsdr:

13.427.1.1 https://git.osmocom.org/rtl-sdr

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file rtl_tcp_commands.h.

13.428 rtl_tcp_commands.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtl_tcp_commands.h
00003  * \brief Defines structures and constants for communicating with rtl_tcp
00004  * \author Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au
00005  *
00006  * This file contains information taken from librtlsdr:
00007  *  https://git.osmocom.org/rtl-sdr
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_RTL_TCP_COMMANDS_H
00020 #define GNSS_SDR_RTL_TCP_COMMANDS_H
00021
00022 #include <boost/asio/ip/tcp.hpp>        // for tcp, tcp::socket
00023 #include <boost/system/error_code.hpp>  // for error_code
00024
00025 /** \addtogroup Signal_Source
00026  * \{ */
00027 /** \addtogroup Signal_Source_libs
00028  * \{ */
00029
00030
00031 /// Command IDs for configuration rtl_tcp
00032 enum RTL_TCP_COMMAND
00033 {
00034     RTL_TCP_SET_FREQUENCY = 1,
00035     RTL_TCP_SET_SAMPLE_RATE = 2,
00036     RTL_TCP_SET_GAIN_MODE = 3,
00037     RTL_TCP_SET_GAIN = 4,
00038     RTL_TCP_SET_IF_GAIN = 6,
00039     RTL_TCP_SET_AGC_MODE = 8
00040 };
00041
00042
00043 /*!
00044  * \brief Send a command to rtl_tcp over the given socket.
00045  */
00046 boost::system::error_code rtl_tcp_command(RTL_TCP_COMMAND id, unsigned param,
00047     boost::asio::ip::tcp::socket &socket);
00048
00049
00050 /** \} */
00051 /** \} */
00052 #endif  // GNSS_SDR_RTL_TCP_COMMANDS_H
```

13.429 rtl_tcp_dongle_info.h File Reference

Interface for a structure sent by rtl_tcp defining the hardware.

```
#include <boost/asio/ip/tcp.hpp>
```

Classes

- class Rtl_Tcp_Dongle_Info

  This class represents the dongle information which is sent by rtl_tcp.

13.429.1 Detailed Description

Interface for a structure sent by rtl_tcp defining the hardware.

Author

Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au

This file contains information taken from librtlsdr:

13.429.1.1 https://git.osmocom.org/rtl-sdr

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file rtl_tcp_dongle_info.h.

13.430 rtl_tcp_dongle_info.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file rtl_tcp_dongle_info.h
00003  * \brief Interface for a structure sent by rtl_tcp defining the hardware.
00004  * \author Anthony Arnold, 2015. anthony.arnold(at)uqconnect.edu.au
00005  *
00006  * This file contains information taken from librtlsdr:
00007  *  https://git.osmocom.org/rtl-sdr
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_RTL_TCP_DONGLE_INFO_H
00020 #define GNSS_SDR_RTL_TCP_DONGLE_INFO_H
00021
00022 #include <boost/asio/ip/tcp.hpp>
00023
00024
00025 /** \addtogroup Signal_Source
00026  * \{ */
00027 /** \addtogroup Signal_Source_libs
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief This class represents the dongle information
00033  * which is sent by rtl_tcp.
00034  */
00035 class Rtl_Tcp_Dongle_Info
00036 {
00037 public:
00038     enum
00039     {
00040         TUNER_UNKNOWN = 0,
00041         TUNER_E4000,
00042         TUNER_FC0012,
00043         TUNER_FC0013,
00044         TUNER_FC2580,
00045         TUNER_R820T,
00046         TUNER_R828D
00047     };
00048
00049     Rtl_Tcp_Dongle_Info();
00050
00051     boost::system::error_code read(boost::asio::ip::tcp::socket &socket);
00052
00053     bool is_valid() const;
00054
00055     const char *get_type_name() const;
00056
00057     double clip_gain(int gain) const;
00058
00059     inline uint32_t get_tuner_type() const
00060     {
00061         return tuner_type_;
```

```
00062     }
00063
00064     inline uint32_t get_tuner_gain_count() const
00065     {
00066         return tuner_gain_count_;
00067     }
00068
00069 private:
00070     char magic_[4]{};
00071     uint32_t tuner_type_;
00072     uint32_t tuner_gain_count_;
00073 };
00074
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_RTL_TCP_DONGLE_INFO_H
```

13.431 telemetry_decoder_adapter.h File Reference

Common functionality for telemetry decoder adapters.

```
#include "telemetry_decoder_interface.h"
#include "telemetry_impl_interface.h"
```

Classes

- class TelemetryDecoderAdapter

  Base class for Telemetry Decoder adapters.

13.431.1 Detailed Description

Common functionality for telemetry decoder adapters.

Author

Carles Fernandez-Prades, 2025 cfernandez@cttc.es Mathieu Favreau, 2026 favreau.mathieu(at)hotmail.↩ com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file telemetry_decoder_adapter.h.

13.432 telemetry_decoder_adapter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file telemetry_decoder_adapter.h
00003  * \brief Common functionality for telemetry decoder adapters
00004  * \author Carles Fernandez-Prades, 2025 cfernandez@cttc.es
00005  *         Mathieu Favreau, 2026 favreau.mathieu(at)hotmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_TELEMETRY_DECODER_ADAPTER_BASE_H
00019 #define GNSS_SDR_TELEMETRY_DECODER_ADAPTER_BASE_H
00020
00021 #include "telemetry_decoder_interface.h"
00022 #include "telemetry_impl_interface.h"
00023
00024 class ConfigurationInterface;
00025
00026 /** \addtogroup Telemetry_Decoder
```

```
00027  * \{
00028  */
00029 /** \addtogroup Telemetry_Decoder_adapters
00030  * \{
00031  */
00032
00033 /*!
00034  * \brief Base class for Telemetry Decoder adapters
00035  */
00036 class TelemetryDecoderAdapter : public TelemetryDecoderInterface
00037 {
00038 public:
00039     TelemetryDecoderAdapter(
00040         const std::string& role,
00041         const std::string& implementation,
00042         unsigned int in_streams,
00043         unsigned int out_streams,
00044         telemetry_impl_interface_sptr decoder);
00045
00046     ~TelemetryDecoderAdapter() override = default;
00047
00048     void connect(gr::top_block_sptr top_block) override;
00049
00050     void disconnect(gr::top_block_sptr top_block) override;
00051
00052     gr::basic_block_sptr get_left_block() override;
00053
00054     gr::basic_block_sptr get_right_block() override;
00055
00056     void set_satellite(const Gnss_Satellite& satellite) override;
00057
00058     std::string role() override;
00059
00060     std::string implementation() override;
00061
00062     void set_channel(int channel) override;
00063
00064     void reset() override;
00065
00066     size_t item_size() override;
00067
00068 private:
00069     telemetry_impl_interface_sptr telemetry_decoder_;
00070     const std::string role_;
00071     const std::string implementation_;
00072 };
00073
00074 /** \} */
00075 /** \} */
00076
00077 #endif  // GNSS_SDR_TELEMETRY_DECODER_ADAPTER_BASE_H
```

13.433 beidou_b1c_telemetry_decoder_gs.h File Reference

BeiDou B1C B-CNAV1 telemetry decoder block.

```
#include "beidou_cnav1_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include "tlm_conf.h"
#include <vector>
```

Classes

- class beidou_b1c_telemetry_decoder_gs

Typedefs

- using beidou_b1c_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b1c_telemetry_decoder_gs>

Functions

- beidou_b1c_telemetry_decoder_gs_sptr **beidou_b1c_make_telemetry_decoder_gs** (const Gnss_Satellite &satellite, const Tlm_Conf &conf)

13.433.1 Detailed Description

BeiDou B1C B-CNAV1 telemetry decoder block.

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_b1c_telemetry_decoder_gs.h.

13.433.2 Typedef Documentation

13.433.2.1 beidou_b1c_telemetry_decoder_gs_sptr

```
using beidou_b1c_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b1c_telemetry_decoder_gs>
```

Definition at line 27 of file beidou_b1c_telemetry_decoder_gs.h.

13.434 beidou_b1c_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b1c_telemetry_decoder_gs.h
00003  * \brief BeiDou B1C B-CNAV1 telemetry decoder block
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_B1C_TELEMETRY_DECODER_GS_H
00017 #define GNSS_SDR_BEIDOU_B1C_TELEMETRY_DECODER_GS_H
00018
00019 #include "beidou_cnav1_navigation_message.h"
00020 #include "nav_message_packet.h"
00021 #include "telemetry_impl_interface.h"
00022 #include "tlm_conf.h"
00023 #include <vector>
00024
00025 class Gnss_Synchro;
00026 class beidou_b1c_telemetry_decoder_gs;
00027 using beidou_b1c_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b1c_telemetry_decoder_gs>;
00028
00029 beidou_b1c_telemetry_decoder_gs_sptr beidou_b1c_make_telemetry_decoder_gs(
00030     const Gnss_Satellite& satellite,
00031     const Tlm_Conf& conf);
00032
00033 class beidou_b1c_telemetry_decoder_gs : public telemetry_impl_interface
00034 {
00035 public:
00036     ~beidou_b1c_telemetry_decoder_gs() override;
00037     void set_satellite(const Gnss_Satellite& satellite) override;
00038     void set_channel(int channel) override;
00039     void reset() override;
00040     void forecast(int noutput_items, gr_vector_int& ninput_items_required) override;
00041     int general_work(int noutput_items, gr_vector_int& ninput_items,
00042         gr_vector_const_void_star& input_items, gr_vector_void_star& output_items) override;
00043
00044 private:
00045     friend beidou_b1c_telemetry_decoder_gs_sptr beidou_b1c_make_telemetry_decoder_gs(
00046         const Gnss_Satellite& satellite, const Tlm_Conf& conf);
00047
00048     beidou_b1c_telemetry_decoder_gs(const Gnss_Satellite& satellite, const Tlm_Conf& conf);
00049     bool decode_frame_from_window(const Gnss_Synchro* window, int32_t start_offset, bool invert);
00050     void publish_navigation(double cn0_db_hz);
00051
00052     Beidou_Cnav1_Navigation_Message d_nav;
00053     Nav_Message_Packet d_nav_msg_packet;
00054     Gnss_Satellite d_satellite;
00055     std::string d_dump_filename;
00056     std::ofstream d_dump_file;
00057     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
```

```
00058     uint64_t d_sample_counter{};
00059     uint64_t d_frame_sync_index{};
00060     int32_t d_channel{};
00061     int32_t d_stat{};
00062     int32_t d_CRC_error_counter{};
00063     int32_t d_frame_soh_offset{0};
00064     uint32_t d_TOW_at_current_symbol_ms{};
00065     uint32_t d_symbol_duration_ms{};
00066     bool d_flag_valid_word{false};
00067     bool d_flag_frame_sync{false};
00068     bool d_flag_PLL_180_deg_phase_locked{false};
00069     bool d_dump{false};
00070     bool d_dump_mat{false};
00071     bool d_remove_dat{false};
00072     bool d_enable_navdata_monitor{false};
00073     bool d_dump_crc_stats{false};
00074     bool d_tow_to_trk{false};
00075     bool d_prev_valid_symbol_output{false};
00076     bool d_await_post_lock_frame_decode{false};
00077     uint32_t d_post_lock_valid_symbols{0U};
00078 };
00079
00080 #endif
```

13.435 beidou_b2a_telemetry_decoder_gs.h File Reference

BeiDou B2a B-CNAV2 telemetry decoder.

```
#include "Beidou_CNAV2.h"
#include "beidou_cnav2_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include "tlm_conf.h"
#include <array>
#include <deque>
#include <vector>
```

Classes

- class beidou_b2a_telemetry_decoder_gs

Typedefs

- using beidou_b2a_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b2a_telemetry_decoder_gs>

Functions

- beidou_b2a_telemetry_decoder_gs_sptr **beidou_b2a_make_telemetry_decoder_gs** (const Gnss_Satellite &satellite, const Tlm_Conf &conf)

13.435.1 Detailed Description

BeiDou B2a B-CNAV2 telemetry decoder.

Author

huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_b2a_telemetry_decoder_gs.h.

13.435.2 Typedef Documentation

13.435.2.1 beidou_b2a_telemetry_decoder_gs_sptr

```
using beidou_b2a_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b2a_telemetry_decoder_gs>
```

Definition at line 30 of file beidou_b2a_telemetry_decoder_gs.h.

13.436 beidou_b2a_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_b2a_telemetry_decoder_gs.h
00003  * \brief BeiDou B2a B-CNAV2 telemetry decoder.
00004  * \author huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_B2A_TELEMETRY_DECODER_GS_H
00017 #define GNSS_SDR_BEIDOU_B2A_TELEMETRY_DECODER_GS_H
00018
00019 #include "Beidou_CNAV2.h"
00020 #include "beidou_cnav2_navigation_message.h"
00021 #include "nav_message_packet.h"
00022 #include "telemetry_impl_interface.h"
00023 #include "tlm_conf.h"
00024 #include <array>
00025 #include <deque>
00026 #include <vector>
00027
00028 class Gnss_Satellite;
00029 class beidou_b2a_telemetry_decoder_gs;
00030 using beidou_b2a_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_b2a_telemetry_decoder_gs>;
00031
00032 beidou_b2a_telemetry_decoder_gs_sptr beidou_b2a_make_telemetry_decoder_gs(
00033     const Gnss_Satellite& satellite,
00034     const Tlm_Conf& conf);
00035
00036 class beidou_b2a_telemetry_decoder_gs : public telemetry_impl_interface
00037 {
00038 public:
00039     ~beidou_b2a_telemetry_decoder_gs() override;
00040     void set_satellite(const Gnss_Satellite& satellite) override;
00041     void set_channel(int channel) override;
00042     void reset() override;
00043     void forecast(int noutput_items, gr_vector_int& ninput_items_required) override;
00044     int general_work(int noutput_items, gr_vector_int& ninput_items,
00045         gr_vector_const_void_star& input_items, gr_vector_void_star& output_items) override;
00046
00047 private:
00048     friend beidou_b2a_telemetry_decoder_gs_sptr beidou_b2a_make_telemetry_decoder_gs(
00049         const Gnss_Satellite& satellite, const Tlm_Conf& conf);
00050     beidou_b2a_telemetry_decoder_gs(const Gnss_Satellite& satellite, const Tlm_Conf& conf);
00051     void publish_navigation(double cn0_db_hz);
00052     bool try_decode_frame();
00053     bool preamble_detected() const;
00054
00055     Beidou_Cnav2_Navigation_Message d_nav;
00056     Nav_Message_Packet d_nav_msg_packet;
00057     Gnss_Satellite d_satellite;
00058     std::deque<float> d_symbol_history;
00059     std::string d_dump_filename;
00060     std::ofstream d_dump_file;
00061     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00062     uint64_t d_sample_counter{};
00063     uint64_t d_preamble_index{};
00064     int32_t d_channel{};
00065     int32_t d_CRC_error_counter{};
00066     uint32_t d_TOW_at_current_symbol_ms{};
00067     bool d_flag_frame_sync{false};
00068     bool d_flag_valid_word{false};
00069     bool d_dump{false};
00070     bool d_dump_mat{false};
00071     bool d_remove_dat{false};
00072     bool d_enable_navdata_monitor{false};
00073     bool d_dump_crc_stats{false};
00074     bool d_tow_to_trk{false};
00075 };
00076
00077 #endif  // GNSS_SDR_BEIDOU_B2A_TELEMETRY_DECODER_GS_H
```

13.437 beidou_dnav_telemetry_decoder_gs.h File Reference

Implementation of a BeiDou DNAV data decoder block for the B1I and B3I signals.

```
#include "beidou_dnav_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <boost/circular_buffer.hpp>
#include <cstdint>
```

Classes

- class beidou_dnav_telemetry_decoder_gs

  This class implements a block that decodes the BeiDou DNAV data, broadcast by both the B1I and B3I signals (frequency_band = 1 or 3, respectively).

Typedefs

- using beidou_dnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_dnav_telemetry_decoder_gs>

Functions

- beidou_dnav_telemetry_decoder_gs_sptr **beidou_dnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)

13.437.1 Detailed Description

Implementation of a BeiDou DNAV data decoder block for the B1I and B3I signals.
Code added as part of GSoC 2018 program.

Author

Damian Miralles, 2018-2019. dmiralles2009(at)gmail.com

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.es

Carles Fernandez, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_dnav_telemetry_decoder_gs.h.

13.438 beidou_dnav_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_telemetry_decoder_gs.h
00003  * \brief Implementation of a BeiDou DNAV data decoder block for the B1I and
00004  * B3I signals
00005  * \details Code added as part of GSoC 2018 program.
00006  * \author Damian Miralles, 2018-2019. dmiralles2009(at)gmail.com
00007  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.es
00008  * \author Carles Fernandez, 2026. cfernandez(at)cttc.es
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_BEIDOU_DNAV_TELEMETRY_DECODER_GS_H
00022 #define GNSS_SDR_BEIDOU_DNAV_TELEMETRY_DECODER_GS_H
00023
00024
```

```
00025 #include "beidou_dnav_navigation_message.h"
00026 #include "nav_message_packet.h"
00027 #include "telemetry_impl_interface.h"
00028 #include <boost/circular_buffer.hpp>
00029 #include <cstdint>
00030
00031
00032 /** \addtogroup Telemetry_Decoder
00033  * \{ */
00034 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00035  * \{ */
00036
00037
00038 class beidou_dnav_telemetry_decoder_gs;
00039
00040 using beidou_dnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<beidou_dnav_telemetry_decoder_gs>;
00041
00042 beidou_dnav_telemetry_decoder_gs_sptr beidou_dnav_make_telemetry_decoder_gs(const Tlm_Conf &conf,
      int32_t frequency_band);
00043
00044
00045 /*!
00046  * \brief This class implements a block that decodes the BeiDou DNAV data,
00047  * broadcast by both the B1I and B3I signals (frequency_band = 1 or 3,
00048  * respectively).
00049  * \note Code added as part of GSoC 2018 program
00050  */
00051 class beidou_dnav_telemetry_decoder_gs : public telemetry_impl_interface
00052 {
00053 public:
00054     ~beidou_dnav_telemetry_decoder_gs() override;                  //!< Class destructor
00055     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00056     void set_channel(int channel) override;                        //!< Set receiver's channel
00057     void reset() override;
00058
00059     /*!
00060      * \brief This is where all signal processing takes place
00061      */
00062     int general_work(int noutput_items, gr_vector_int &ninput_items,
00063         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00064
00065 private:
00066     friend beidou_dnav_telemetry_decoder_gs_sptr beidou_dnav_make_telemetry_decoder_gs(const Tlm_Conf
      &conf, int32_t frequency_band);
00067
00068     beidou_dnav_telemetry_decoder_gs(const Tlm_Conf &conf, int32_t frequency_band);
00069
00070     void decode_subframe(float *symbols, double cn0);
00071     void decode_word(int32_t word_counter, const float *enc_word_symbols, int32_t *dec_word_symbols);
00072     void decode_bch15_11_01(const int32_t *bits, std::array<int32_t, 15> &decbits);
00073
00074     // Preamble decoding
00075     std::array<int32_t, BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS> d_preamble_samples{};
00076
00077     std::array<float, BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS> d_subframe_symbols{};
00078
00079     // Storage for incoming data
00080     boost::circular_buffer<float> d_symbol_history;
00081
00082     // Navigation Message variable
00083     Beidou_Dnav_Navigation_Message d_nav;
00084
00085     Nav_Message_Packet d_nav_msg_packet;
00086     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00087
00088     // Satellite Information and logging capacity
00089     Gnss_Satellite d_satellite;
00090     std::string d_dump_filename;
00091     std::ofstream d_dump_file;
00092
00093     uint64_t d_sample_counter;  // Sample counter as an index (1,2,3,..etc) indicating number of
      samples processed
00094     uint64_t d_preamble_index;  // Index of sample number where preamble was found
00095
00096     int32_t d_channel;
00097     int32_t d_symbols_per_preamble;
00098     int32_t d_samples_per_preamble;
00099     int32_t d_preamble_period_samples;
00100     int32_t d_CRC_error_counter;  // Number of failed CRC operations
00101     const int32_t d_signal_type;  // DNAV data source (1:B1I, 5:B3I)
00102     uint32_t d_required_symbols;
00103     uint32_t d_stat;  // Status of decoder
00104
00105     // Values to populate gnss synchronization structure
00106     uint64_t d_last_valid_preamble;
00107     const uint32_t d_symbol_duration_d1_ms;      // D1 NAV message (MEO/IGSO satellites)
00108     const uint32_t d_symbol_duration_geo_d2_ms;  // D2 NAV message (GEO satellites)
```

```
00109     uint32_t d_symbol_duration_ms;
00110     uint32_t d_TOW_at_Preamble_ms;
00111     uint32_t d_TOW_at_current_symbol_ms;
00112
00113     const char d_band;  // Frequency band the block is configured for ('1' for B1I, '3' for B3I)
00114
00115     bool d_flag_SOW_set;     // Indicates when time of week is set
00116     bool d_flag_frame_sync;  // Indicate when a frame sync is achieved
00117     bool d_flag_preamble;    // Flag indicating when preamble was found
00118
00119     bool d_flag_valid_word;
00120     bool d_sent_tlm_failed_msg;
00121     const bool d_dump;
00122     const bool d_dump_mat;
00123     const bool d_remove_dat;
00124     const bool d_enable_navdata_monitor;
00125     bool d_dump_crc_stats;
00126     const bool d_tow_to_trk;
00127 };
00128
00129
00130 /** \} */
00131 /** \} */
00132 #endif  // GNSS_SDR_BEIDOU_DNAV_TELEMETRY_DECODER_GS_H
```

13.439 galileo_telemetry_decoder_gs.h File Reference

Implementation of a Galileo unified INAV and FNAV message demodulator block.

```
#include "galileo_cnav_message.h"
#include "galileo_fnav_message.h"
#include "galileo_inav_message.h"
#include "gnss_time.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <boost/circular_buffer.hpp>
```

Classes

- class galileo_telemetry_decoder_gs

  This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD.

Typedefs

- using galileo_telemetry_decoder_gs_sptr = gnss_shared_ptr<galileo_telemetry_decoder_gs>

Functions

- galileo_telemetry_decoder_gs_sptr **galileo_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int frame_type)

13.439.1 Detailed Description

Implementation of a Galileo unified INAV and FNAV message demodulator block.

Author

Javier Arribas 2018. jarribas(at)cttc.es

Carles Fernandez, 2021-2022. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_telemetry_decoder_gs.h.

13.440 galileo_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_telemetry_decoder_gs.h
00003  * \brief Implementation of a Galileo unified INAV and FNAV message demodulator
00004  * block
00005  * \author Javier Arribas 2018. jarribas(at)cttc.es
00006  * \author Carles Fernandez, 2021-2022. cfernandez(at)cttc.es
00007  *
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020
00021 #ifndef GNSS_SDR_GALILEO_TELEMETRY_DECODER_GS_H
00022 #define GNSS_SDR_GALILEO_TELEMETRY_DECODER_GS_H
00023
00024 #include "galileo_cnav_message.h"  // for Galileo_Cnav_Message
00025 #include "galileo_fnav_message.h"  // for Galileo_Fnav_Message
00026 #include "galileo_inav_message.h"  // for Galileo_Inav_Message
00027 #include "gnss_time.h"
00028 #include "nav_message_packet.h"
00029 #include "telemetry_impl_interface.h"  // for telemetry_impl_interface
00030 #include <boost/circular_buffer.hpp>   // for boost::circular_buffer
00031
00032
00033 /** \addtogroup Telemetry_Decoder
00034  * \{ */
00035 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00036  * \{ */
00037
00038 class Viterbi_Decoder;               // forward declaration
00039 class Gnss_Synchro;                  // forward declaration
00040 class galileo_telemetry_decoder_gs;  // forward declaration
00041
00042 using galileo_telemetry_decoder_gs_sptr = gnss_shared_ptr<galileo_telemetry_decoder_gs>;
00043
00044 galileo_telemetry_decoder_gs_sptr galileo_make_telemetry_decoder_gs(const Tlm_Conf &conf, int
      frame_type);
00045
00046 /*!
00047  * \brief This class implements a block that decodes the INAV and FNAV data defined in Galileo ICD
00048  */
00049 class galileo_telemetry_decoder_gs : public telemetry_impl_interface
00050 {
00051 public:
00052     ~galileo_telemetry_decoder_gs() override;
00053     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00054     void set_channel(int32_t channel) override;                    //!< Set receiver's channel
00055     void reset() override;
00056
00057     /*!
00058      * \brief This is where all signal processing takes place
00059      */
00060     int general_work(int noutput_items, gr_vector_int &ninput_items,
00061         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00062
00063 private:
00064     friend galileo_telemetry_decoder_gs_sptr galileo_make_telemetry_decoder_gs(const Tlm_Conf &conf,
      int frame_type);
00065
00066     galileo_telemetry_decoder_gs(const Tlm_Conf &conf, int frame_type);
00067
00068     struct CnavPageReceptionTime
00069     {
00070         uint32_t week;
00071         uint32_t tow_ms;
00072     };
00073
00074     // Callers must hold d_setlock while resetting or checking decoder state.
00075     void reset_decoder_state();
00076     void check_tlm_separation();
00077
00078     void msg_handler_read_galileo_tow_map(const pmt::pmt_t &msg);
00079     void clear_galileo_tow_map_entry();
00080     void publish_galileo_tow_map_entry(uint32_t week, uint32_t tow_ms, uint64_t sample_counter);
00081     void publish_current_galileo_tow_map_entry(uint64_t sample_counter);
```

```
00082     bool update_known_galileo_week(int32_t week);
00083     bool set_current_tow_from_preamble(uint32_t preamble_week, uint32_t preamble_tow_ms, int64_t
      delay_ms);
00084     void advance_current_tow(int64_t delta_ms);
00085     void capture_pending_reduced_ced(double cn0);
00086     void publish_pending_reduced_ced();
00087     CnavPageReceptionTime get_cnav_page_reception_time(const Gnss_Synchro &current_symbol) const;
00088     void deinterleaver(int32_t rows, int32_t cols, const float *in, float *out);
00089     void decode_INAV_word(float *page_part_symbols, int32_t frame_length, double cn0);
00090     void decode_FNAV_word(float *page_symbols, int32_t frame_length, double cn0);
00091     void decode_CNAV_word(uint64_t time_stamp, CnavPageReceptionTime page_reception_time, float
      *page_symbols, int32_t page_length, double cn0);
00092
00093     std::unique_ptr<Viterbi_Decoder> d_viterbi;
00094     std::vector<int32_t> d_preamble_samples;
00095     std::vector<float> d_page_part_symbols;
00096
00097     std::string d_dump_filename;
00098     std::ofstream d_dump_file;
00099
00100     boost::circular_buffer<float> d_symbol_history;
00101
00102     Gnss_Satellite d_satellite;
00103
00104     // navigation message vars
00105     Galileo_Cnav_Message d_cnav_nav;
00106     Galileo_Inav_Message d_inav_nav;
00107     Galileo_Fnav_Message d_fnav_nav;
00108     Galileo_Reduced_CED d_pending_reduced_ced_data;
00109
00110     Nav_Message_Packet d_nav_msg_packet;
00111     GnssTime d_current_timetag{};
00112
00113     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00114
00115     double d_delta_t;  // GPS-GALILEO time offset
00116
00117     uint64_t d_symbol_counter;
00118     uint64_t d_preamble_index;
00119     uint64_t d_last_valid_preamble;
00120     uint64_t d_received_sample_counter;
00121     uint64_t d_pending_reduced_ced_start_symbol;
00122
00123     double d_pending_reduced_ced_cn0;
00124
00125     int32_t d_mm;
00126     int32_t d_codelength;
00127     int32_t d_datalength;
00128     const int32_t d_frame_type;
00129     int32_t d_bits_per_preamble;
00130     int32_t d_samples_per_preamble;
00131     int32_t d_preamble_period_symbols;
00132     int32_t d_CRC_error_counter;
00133     int32_t d_channel;
00134     int32_t d_flag_even_word_arrived;
00135
00136     uint32_t d_PRN_code_period_ms;
00137     uint32_t d_required_symbols;
00138     uint32_t d_frame_length_symbols;
00139     uint32_t d_stat;
00140     uint32_t d_TOW_at_Preamble_ms;
00141     uint32_t d_TOW_at_current_symbol_ms;
00142     uint32_t d_max_symbols_without_valid_frame;
00143     uint32_t d_received_week;
00144     uint32_t d_received_tow_ms;
00145     uint32_t d_TOW_week;
00146     uint32_t d_galileo_week;
00147
00148     char d_band;  // This variable will store which band we are dealing with (Galileo E1 or E5b)
00149
00150     bool d_sent_tlm_failed_msg;
00151     bool d_flag_frame_sync;
00152     bool d_flag_PLL_180_deg_phase_locked;
00153     bool d_flag_preamble;
00154     const bool d_dump;
00155     const bool d_dump_mat;
00156     const bool d_remove_dat;
00157     bool d_first_eph_sent;
00158     bool d_pending_reduced_ced;
00159     bool d_cnav_dummy_page;
00160     bool d_print_cnav_page;
00161     const bool d_enable_navdata_monitor;
00162     bool d_dump_crc_stats;
00163     bool d_enable_reed_solomon_inav;
00164     bool d_valid_timetag;
00165     bool d_E6_TOW_set;
00166     bool d_galileo_week_valid;
```

```
00167     const bool d_there_are_e1_channels;
00168     const bool d_there_are_e6_channels;
00169     const bool d_use_ced;
00170     const bool d_tow_to_trk;
00171 };
00172
00173
00174 /** \} */
00175 /** \} */
00176 #endif  // GNSS_SDR_GALILEO_TELEMETRY_DECODER_GS_H
```

13.441 glonass_gnav_telemetry_decoder_gs.h File Reference

Implementation of a GLONASS GNAV data decoder block for the L1 and L2 C/A signals.

```
#include "GLONASS_L1_L2_CA.h"
#include "glonass_gnav_navigation_message.h"
#include "gnss_synchro.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <boost/circular_buffer.hpp>
#include <cstdint>
```

Classes

- class glonass_gnav_telemetry_decoder_gs

  This class implements a block that decodes the GNAV data defined in GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals (frequency_band = 1 or 2, respectively).

Typedefs

- using glonass_gnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<glonass_gnav_telemetry_decoder_gs>

Functions

- glonass_gnav_telemetry_decoder_gs_sptr **glonass_gnav_make_telemetry_decoder_gs** (const Tlm_Conf &conf, int32_t frequency_band)

13.441.1 Detailed Description

Implementation of a GLONASS GNAV data decoder block for the L1 and L2 C/A signals.

Note

Code added as part of GSoC 2017 program

Author

Damian Miralles, 2017-2018. dmiralles2009(at)gmail.com

Carles Fernandez, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_gnav_telemetry_decoder_gs.h.

13.442 glonass_gnav_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_gnav_telemetry_decoder_gs.h
00003  * \brief Implementation of a GLONASS GNAV data decoder block for the L1 and
00004  * L2 C/A signals
00005  * \note Code added as part of GSoC 2017 program
```

```
00006  * \author Damian Miralles, 2017-2018. dmiralles2009(at)gmail.com
00007  * \author Carles Fernandez, 2026. cfernandez(at)cttc.es
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_GLONASS_GNAV_TELEMETRY_DECODER_GS_H
00021 #define GNSS_SDR_GLONASS_GNAV_TELEMETRY_DECODER_GS_H
00022
00023
00024 #include "GLONASS_L1_L2_CA.h"
00025 #include "glonass_gnav_navigation_message.h"
00026 #include "gnss_synchro.h"
00027 #include "nav_message_packet.h"
00028 #include "telemetry_impl_interface.h"
00029 #include <boost/circular_buffer.hpp>
00030 #include <cstdint>
00031
00032 /** \addtogroup Telemetry_Decoder
00033  * \{ */
00034 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00035  * \{ */
00036
00037 class glonass_gnav_telemetry_decoder_gs;
00038
00039 using glonass_gnav_telemetry_decoder_gs_sptr = gnss_shared_ptr<glonass_gnav_telemetry_decoder_gs>;
00040
00041 glonass_gnav_telemetry_decoder_gs_sptr glonass_gnav_make_telemetry_decoder_gs(const Tlm_Conf &conf,
      int32_t frequency_band);
00042
00043 /*!
00044  * \brief This class implements a block that decodes the GNAV data defined in
00045  * GLONASS ICD v5.1, broadcast by both the L1 and L2 C/A signals
00046  * (frequency_band = 1 or 2, respectively).
00047  * \note Code added as part of GSoC 2017 program
00048  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00049  *
00050  */
00051 class glonass_gnav_telemetry_decoder_gs : public telemetry_impl_interface
00052 {
00053 public:
00054     ~glonass_gnav_telemetry_decoder_gs() override;                 //!< Class destructor
00055     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00056     void set_channel(int32_t channel) override;                    //!< Set receiver's channel
00057     void reset() override;                                         //!< Clear synchronization state on
      channel re-acquisition
00058
00059     /*!
00060      * \brief This is where all signal processing takes place
00061      */
00062     int general_work(int noutput_items, gr_vector_int &ninput_items,
00063         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00064
00065 private:
00066     friend glonass_gnav_telemetry_decoder_gs_sptr glonass_gnav_make_telemetry_decoder_gs(const
      Tlm_Conf &conf, int32_t frequency_band);
00067
00068     glonass_gnav_telemetry_decoder_gs(const Tlm_Conf &conf, int32_t frequency_band);
00069
00070     const std::array<int16_t, GLONASS_GNAV_PREAMBLE_LENGTH_BITS>
      d_preambles_bits{GLONASS_GNAV_PREAMBLE_SAMPLES};
00071
00072     void decode_string(const double *symbols, double cn0);
00073
00074     // Storage for incoming data
00075     boost::circular_buffer<Gnss_Synchro> d_symbol_history;
00076
00077     // Navigation Message variable
00078     Glonass_Gnav_Navigation_Message d_nav;
00079
00080     Gnss_Satellite d_satellite;
00081
00082     Nav_Message_Packet d_nav_msg_packet;
00083     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00084
00085     std::string d_dump_filename;
00086     std::ofstream d_dump_file;
```

```
00087
00088     double d_TOW_at_current_symbol;
00089     const double d_symbol_period_s;
00090
00091     // Variables for internal functionality
00092     uint64_t d_sample_counter;  // Sample counter as an index (1,2,3,..etc) indicating number of
      samples processed
00093     uint64_t d_preamble_index;  // Index of sample number where preamble was found
00094
00095     uint32_t d_stat;              // Status of decoder
00096     int32_t d_CRC_error_counter;  // Number of failed CRC operations
00097     int32_t d_channel;
00098
00099     const char d_band;  // Frequency band the block is configured for ('1' or '2')
00100
00101     bool d_flag_frame_sync;  // Indicate when a frame sync is achieved
00102     bool d_flag_preamble;    // Flag indicating when preamble was found
00103     const bool d_dump;
00104     const bool d_dump_mat;
00105     const bool d_remove_dat;
00106     const bool d_enable_navdata_monitor;
00107     bool d_dump_crc_stats;
00108     const bool d_tow_to_trk;
00109 };
00110
00111
00112 /** \} */
00113 /** \} */
00114 #endif  // GNSS_SDR_GLONASS_GNAV_TELEMETRY_DECODER_GS_H
```

13.443 gps_l1_ca_telemetry_decoder_gs.h File Reference

Interface of a NAV message demodulator block based on Kay Borre book MATLAB-based GPS receiver.

```
#include "gps_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <boost/circular_buffer.hpp>
#include <cstdint>
```

Classes

- class gps_l1_ca_telemetry_decoder_gs

  This class implements a block that decodes the NAV data defined in IS-GPS-200M.

Typedefs

- using gps_l1_ca_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l1_ca_telemetry_decoder_gs>

Enumerations

- enum class **L1LnavSystem** { **GPS** , **QZSS** }

Functions

- gps_l1_ca_telemetry_decoder_gs_sptr **gps_l1_ca_make_telemetry_decoder_gs** (const Tlm_Conf &conf, L1LnavSystem system=L1LnavSystem::GPS)

13.443.1 Detailed Description

Interface of a NAV message demodulator block based on Kay Borre book MATLAB-based GPS receiver.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Carles Fernandez Prades, 2011-2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_telemetry_decoder_gs.h.

13.444 gps_l1_ca_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_telemetry_decoder_gs.h
00003  * \brief Interface of a NAV message demodulator block based on
00004  * Kay Borre book MATLAB-based GPS receiver
00005  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00006  * \author Carles Fernandez Prades, 2011-2026. cfernandez(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GPS_L1_CA_TELEMETRY_DECODER_GS_H
00020 #define GNSS_SDR_GPS_L1_CA_TELEMETRY_DECODER_GS_H
00021
00022 #include "gps_navigation_message.h"
00023 #include "nav_message_packet.h"
00024 #include "telemetry_impl_interface.h"
00025 #include <boost/circular_buffer.hpp>
00026 #include <cstdint>
00027
00028
00029 /** \addtogroup Telemetry_Decoder
00030  * \{ */
00031 /** \addtogroup Telemetry_Decoder_gnuradio_blocks telemetry_decoder_gr_blocks
00032  * GNU Radio blocks for the demodulation of GNSS navigation messages.
00033  * \{ */
00034
00035 enum class L1LnavSystem
00036 {
00037     GPS,
00038     QZSS
00039 };
00040
00041 class Gnss_Synchro;
00042 class gps_l1_ca_telemetry_decoder_gs;
00043
00044 using gps_l1_ca_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l1_ca_telemetry_decoder_gs>;
00045
00046 gps_l1_ca_telemetry_decoder_gs_sptr gps_l1_ca_make_telemetry_decoder_gs(
00047     const Tlm_Conf &conf,
00048     L1LnavSystem system = L1LnavSystem::GPS);
00049
00050 /*!
00051  * \brief This class implements a block that decodes the NAV data defined in IS-GPS-200M
00052  */
00053 class gps_l1_ca_telemetry_decoder_gs : public telemetry_impl_interface
00054 {
00055 public:
00056     ~gps_l1_ca_telemetry_decoder_gs() override;
00057     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00058     void set_channel(int channel) override;                        //!< Set receiver's channel
00059     void reset() override;
00060
00061     /*!
00062      * \brief This is where all signal processing takes place
00063      */
00064     int general_work(int noutput_items, gr_vector_int &ninput_items,
00065         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00066
00067 private:
00068     friend gps_l1_ca_telemetry_decoder_gs_sptr gps_l1_ca_make_telemetry_decoder_gs(
00069         const Tlm_Conf &conf,
00070         L1LnavSystem system);
00071
00072     gps_l1_ca_telemetry_decoder_gs(const Tlm_Conf &conf, L1LnavSystem system);
00073
00074     void check_tlm_separation();
00075     void frame_synchronization(const Gnss_Synchro &current_gs);
00076     bool is_PLL_180_deg_phase_locked();
00077     bool gps_word_parityCheck(uint32_t gpsword);
00078     bool decode_subframe(double cn0, bool flag_invert);
00079     bool is_tow_consistent(uint32_t decoded_tow_s);
00080
00081     const L1LnavSystem d_system;
00082     std::unique_ptr<Gps_Navigation_Message> d_nav;
00083     Gnss_Satellite d_satellite;
```

```
00084     Nav_Message_Packet d_nav_msg_packet;
00085     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00086
00087     std::array<int32_t, GPS_CA_PREAMBLE_LENGTH_BITS> d_preamble_samples{};
00088
00089     std::string d_dump_filename;
00090     std::ofstream d_dump_file;
00091
00092     boost::circular_buffer<float> d_symbol_history;
00093
00094     uint64_t d_sample_counter;
00095     uint64_t d_preamble_index;
00096     uint64_t d_last_valid_preamble;
00097
00098     const int32_t d_bits_per_preamble;
00099     const int32_t d_samples_per_preamble;
00100     const int32_t d_preamble_period_symbols;
00101     int32_t d_CRC_error_counter;
00102     int32_t d_channel;
00103
00104     const uint32_t d_required_symbols;
00105     uint32_t d_prev_GPS_frame_4bytes;
00106     const uint32_t d_max_symbols_without_valid_frame;
00107     uint32_t d_stat;
00108     uint32_t d_TOW_at_Preamble_ms;
00109     uint32_t d_TOW_at_current_symbol_ms;
00110     uint32_t d_last_decoded_tow_s;
00111     uint64_t d_last_decoded_tow_sample_counter;
00112
00113     bool d_flag_frame_sync;
00114     bool d_flag_preamble;
00115     bool d_sent_tlm_failed_msg;
00116     bool d_flag_PLL_180_deg_phase_locked;
00117     bool d_flag_TOW_set;
00118     const bool d_dump;
00119     const bool d_dump_mat;
00120     const bool d_remove_dat;
00121     const bool d_enable_navdata_monitor;
00122     bool d_dump_crc_stats;
00123     const bool d_tow_to_trk;
00124     bool d_have_last_decoded_tow;
00125     bool d_flag_frame_sync_confirmed;
00126     bool d_flag_subframe_parity_ok;
00127 };
00128
00129
00130 /** \} */
00131 /** \} */
00132 #endif  // GNSS_SDR_GPS_L1_CA_TELEMETRY_DECODER_GS_H
```

13.445 gps_l2c_telemetry_decoder_gs.h File Reference

Interface of a CNAV message demodulator block.

```
#include "gps_cnav_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <cstdint>
#include "cnav_msg.h"
```

Classes

- class gps_l2c_telemetry_decoder_gs

  This class implements a block that decodes CNAV data defined in IS-GPS-200M.

Typedefs

- using gps_l2c_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l2c_telemetry_decoder_gs>

Functions

- gps_l2c_telemetry_decoder_gs_sptr **gps_l2c_make_telemetry_decoder_gs** (const Tlm_Conf &conf)

13.445.1 Detailed Description

Interface of a CNAV message demodulator block.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_l2c_telemetry_decoder_gs.h.

13.446 gps_l2c_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l2c_telemetry_decoder_gs.h
00003  * \brief Interface of a CNAV message demodulator block
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GPS_L2C_TELEMETRY_DECODER_GS_H
00018 #define GNSS_SDR_GPS_L2C_TELEMETRY_DECODER_GS_H
00019
00020
00021 #include "gps_cnav_navigation_message.h"
00022 #include "nav_message_packet.h"
00023 #include "telemetry_impl_interface.h"
00024 #include <cstdint>
00025
00026 extern "C"
00027 {
00028 #include "cnav_msg.h"
00029 }
00030
00031 /** \addtogroup Telemetry_Decoder
00032  * \{ */
00033 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00034  * \{ */
00035
00036
00037 class gps_l2c_telemetry_decoder_gs;
00038
00039 using gps_l2c_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l2c_telemetry_decoder_gs>;
00040
00041 gps_l2c_telemetry_decoder_gs_sptr gps_l2c_make_telemetry_decoder_gs(const Tlm_Conf &conf);
00042
00043 /*!
00044  * \brief This class implements a block that decodes CNAV data defined in IS-GPS-200M
00045  */
00046 class gps_l2c_telemetry_decoder_gs : public telemetry_impl_interface
00047 {
00048 public:
00049     ~gps_l2c_telemetry_decoder_gs() override;
00050     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00051     void set_channel(int32_t channel) override;                    //!< Set receiver's channel
00052     void reset() override;
00053
00054     /*!
00055      * \brief This is where all signal processing takes place
00056      */
00057     int general_work(int noutput_items, gr_vector_int &ninput_items,
00058         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00059
00060 private:
00061     friend gps_l2c_telemetry_decoder_gs_sptr gps_l2c_make_telemetry_decoder_gs(const Tlm_Conf &conf);
00062
00063     explicit gps_l2c_telemetry_decoder_gs(const Tlm_Conf &conf);
00064
00065     Gnss_Satellite d_satellite;
00066
00067     cnav_msg_decoder_t d_cnav_decoder{};
```

```
00068
00069     Gps_CNAV_Navigation_Message d_CNAV_Message;
00070
00071     Nav_Message_Packet d_nav_msg_packet;
00072     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00073
00074     std::string d_dump_filename;
00075     std::ofstream d_dump_file;
00076
00077     double d_TOW_at_current_symbol;
00078     double d_TOW_at_Preamble;
00079
00080     uint64_t d_sample_counter;
00081     uint64_t d_last_valid_preamble;
00082
00083     int32_t d_channel;
00084
00085     const uint32_t d_max_symbols_without_valid_frame;
00086
00087     const bool d_dump;
00088     bool d_sent_tlm_failed_msg;
00089     bool d_flag_PLL_180_deg_phase_locked;
00090     bool d_flag_valid_word;
00091     const bool d_dump_mat;
00092     const bool d_remove_dat;
00093     const bool d_enable_navdata_monitor;
00094     bool d_dump_crc_stats;
00095     const bool d_tow_to_trk;
00096 };
00097
00098
00099 /** \} */
00100 /** \} */
00101 #endif  // GNSS_SDR_GPS_L2C_TELEMETRY_DECODER_GS_H
```

13.447 gps_l5_telemetry_decoder_gs.h File Reference

Interface of a CNAV message demodulator block.

```
#include "gps_cnav_navigation_message.h"
#include "nav_message_packet.h"
#include "telemetry_impl_interface.h"
#include <boost/circular_buffer.hpp>
#include <cstdint>
#include "cnav_msg.h"
```

Classes

- class gps_l5_telemetry_decoder_gs

  This class implements a GPS L5 Telemetry decoder.

Typedefs

- using gps_l5_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l5_telemetry_decoder_gs>

Functions

- gps_l5_telemetry_decoder_gs_sptr **gps_l5_make_telemetry_decoder_gs** (const Tlm_Conf &conf, Cnav↩ System system=CnavSystem::GPS)

13.447.1 Detailed Description

Interface of a CNAV message demodulator block.

Author

Antonio Ramos, 2017. antonio.ramos(at)cttc.es

13.447.1.1 autotoc_md445

Author

Carles Fernandez Prades, 2017-2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_l5_telemetry_decoder_gs.h.

13.448 gps_l5_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l5_telemetry_decoder_gs.h
00003  * \brief Interface of a CNAV message demodulator block
00004  * \author Antonio Ramos, 2017. antonio.ramos(at)cttc.es
00005  * \author Carles Fernandez Prades, 2017-2026. cfernandez(at)cttc.es
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GPS_L5_TELEMETRY_DECODER_GS_H
00018 #define GNSS_SDR_GPS_L5_TELEMETRY_DECODER_GS_H
00019
00020
00021 #include "gps_cnav_navigation_message.h"  // for Gps_CNAV_Navigation_Message
00022 #include "nav_message_packet.h"
00023 #include "telemetry_impl_interface.h"
00024 #include <boost/circular_buffer.hpp>
00025 #include <cstdint>
00026
00027 extern "C"
00028 {
00029 #include "cnav_msg.h"
00030 }
00031
00032 /** \addtogroup Telemetry_Decoder
00033  * \{ */
00034 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00035  * \{ */
00036
00037
00038 class gps_l5_telemetry_decoder_gs;
00039
00040 using gps_l5_telemetry_decoder_gs_sptr = gnss_shared_ptr<gps_l5_telemetry_decoder_gs>;
00041
00042 gps_l5_telemetry_decoder_gs_sptr gps_l5_make_telemetry_decoder_gs(
00043     const Tlm_Conf &conf,
00044     CnavSystem system = CnavSystem::GPS);
00045
00046 /*!
00047  * \brief This class implements a GPS L5 Telemetry decoder
00048  *
00049  */
00050 class gps_l5_telemetry_decoder_gs : public telemetry_impl_interface
00051 {
00052 public:
00053     ~gps_l5_telemetry_decoder_gs() override;
00054     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00055     void set_channel(int32_t channel) override;                    //!< Set receiver's channel
00056     void reset() override;
00057     int general_work(int noutput_items, gr_vector_int &ninput_items,
00058         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00059
00060 private:
00061     friend gps_l5_telemetry_decoder_gs_sptr gps_l5_make_telemetry_decoder_gs(
00062         const Tlm_Conf &conf,
00063         CnavSystem system);
00064
00065     gps_l5_telemetry_decoder_gs(const Tlm_Conf &conf, CnavSystem system);
00066
00067     cnav_msg_decoder_t d_cnav_decoder{};
00068
00069     Gnss_Satellite d_satellite;
```

```
00070     const CnavSystem d_system;
00071
00072     std::unique_ptr<Gps_CNAV_Navigation_Message> d_CNAV_Message;
00073
00074     Nav_Message_Packet d_nav_msg_packet;
00075     std::unique_ptr<Tlm_CRC_Stats> d_Tlm_CRC_Stats;
00076
00077     std::string d_dump_filename;
00078     std::ofstream d_dump_file;
00079
00080     uint64_t d_sample_counter;
00081     uint64_t d_last_valid_preamble;
00082
00083     int32_t d_channel;
00084
00085     uint32_t d_TOW_at_current_symbol_ms;
00086     uint32_t d_TOW_at_Preamble_ms;
00087     const uint32_t d_max_symbols_without_valid_frame;
00088
00089     bool d_flag_PLL_180_deg_phase_locked;
00090     bool d_flag_valid_word;
00091     bool d_sent_tlm_failed_msg;
00092     const bool d_dump;
00093     const bool d_dump_mat;
00094     const bool d_remove_dat;
00095     const bool d_enable_navdata_monitor;
00096     bool d_dump_crc_stats;
00097     const bool d_tow_to_trk;
00098 };
00099
00100
00101 /** \} */
00102 /** \} */
00103 #endif  // GNSS_SDR_GPS_L5_TELEMETRY_DECODER_GS_H
```

13.449 sbas_l1_telemetry_decoder_gs.h File Reference

Interface of a SBAS telemetry data decoder block.

```
#include "telemetry_impl_interface.h"
#include <boost/crc.hpp>
#include <gnuradio/types.h>
#include <cstddef>
#include <deque>
#include <fstream>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class SbasL1DumpSession
- class sbas_l1_telemetry_decoder_gs

  This class implements a block that decodes the SBAS integrity and corrections data defined in RTCA MOPS DO-229.

Typedefs

- using sbas_l1_telemetry_decoder_gs_sptr = gnss_shared_ptr<sbas_l1_telemetry_decoder_gs>

Functions

- sbas_l1_telemetry_decoder_gs_sptr **sbas_l1_make_telemetry_decoder_gs** (bool dump, std::string dump_filename=std::string())

13.449.1 Detailed Description

Interface of a SBAS telemetry data decoder block.

Author

Daniel Fehr 2013. daniel.co(at)bluewin.ch

Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es

Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file sbas_l1_telemetry_decoder_gs.h.

13.450 sbas_l1_telemetry_decoder_gs.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sbas_l1_telemetry_decoder_gs.h
00003  * \brief Interface of a SBAS telemetry data decoder block
00004  * \author Daniel Fehr 2013. daniel.co(at)bluewin.ch
00005  * \author Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es
00006  * \author Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_SBAS_L1_TELEMETRY_DECODER_GS_H
00020 #define GNSS_SDR_SBAS_L1_TELEMETRY_DECODER_GS_H
00021
00022 #include "telemetry_impl_interface.h"
00023 #include <boost/crc.hpp>      // for crc_optimal
00024 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00025 #include <cstddef>           // for size_t
00026 #include <deque>
00027 #include <fstream>
00028 #include <memory>
00029 #include <string>
00030 #include <vector>
00031
00032 /** \addtogroup Telemetry_Decoder
00033  * \{ */
00034 /** \addtogroup Telemetry_Decoder_gnuradio_blocks
00035  * \{ */
00036
00037
00038 class Viterbi_Decoder_Sbas;
00039
00040 class sbas_l1_telemetry_decoder_gs;
00041
00042 class SbasL1DumpSession
00043 {
00044 public:
00045     SbasL1DumpSession();
00046     ~SbasL1DumpSession();
00047
00048     SbasL1DumpSession(const SbasL1DumpSession &) = delete;
00049     SbasL1DumpSession &operator=(const SbasL1DumpSession &) = delete;
00050
00051     //! Opens filename with truncation on first use and append on later uses.
00052     //! Returns true on first use; stream state reports whether opening succeeded.
00053     bool open(std::ofstream &stream, const std::string &filename);
00054
00055 private:
00056     class Impl;
00057     std::unique_ptr<Impl> d_impl;
00058 };
00059
00060 using sbas_l1_telemetry_decoder_gs_sptr = gnss_shared_ptr<sbas_l1_telemetry_decoder_gs>;
00061
00062 sbas_l1_telemetry_decoder_gs_sptr sbas_l1_make_telemetry_decoder_gs(
00063     bool dump,
00064     std::string dump_filename = std::string());
00065
00066 /*!
00067  * \brief This class implements a block that decodes the SBAS integrity and
00068  * corrections data defined in RTCA MOPS DO-229
```

```
00069  */
00070 class sbas_l1_telemetry_decoder_gs : public telemetry_impl_interface
00071 {
00072 public:
00073     ~sbas_l1_telemetry_decoder_gs() override;
00074     void set_satellite(const Gnss_Satellite &satellite) override;  //!< Set satellite PRN
00075     void set_channel(int32_t channel) override;                    //!< Set receiver's channel
00076     //! Clears all internal decoding state (buffers, aligners, Viterbi decoders)
00077     //! and closes the dump file, so that a channel reassigned to a different
00078     //! PRN never mixes messages from the previous satellite into its file.
00079     void reset() override;
00080
00081     //! Exposed for unit testing: appends timestamps entering a continuous
00082     //! decoder and returns the timestamps corresponding to the decoded bits.
00083     static std::vector<double> consume_decoded_timestamps(
00084         std::deque<double> &pending_stamps,
00085         const std::vector<double> &input_stamps,
00086         int32_t decoded_bits);
00087
00088     /*!
00089      * \brief This is where all signal processing takes place
00090      */
00091     int general_work(int noutput_items, gr_vector_int &ninput_items,
00092         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00093
00094 private:
00095     friend sbas_l1_telemetry_decoder_gs_sptr sbas_l1_make_telemetry_decoder_gs(
00096         bool dump,
00097         std::string dump_filename);
00098
00099     explicit sbas_l1_telemetry_decoder_gs(bool dump, std::string dump_filename = std::string());
00100     static std::shared_ptr<SbasL1DumpSession> acquire_dump_session(const std::string &dump_filename);
00101
00102     void viterbi_decoder(double *page_part_symbols, int32_t *page_part_bits);
00103     void align_samples();
00104
00105     static const int32_t D_SAMPLES_PER_SYMBOL_SBAS = 2;
00106     static const int32_t D_SYMBOLS_PER_BIT_SBAS = 2;
00107     static const int32_t D_BLOCK_SIZE_IN_BITS_SBAS = 30;
00108
00109     bool d_dump;
00110     Gnss_Satellite d_satellite;
00111     int32_t d_channel;
00112
00113     std::string d_dump_filename;
00114     std::ofstream d_dump_file;
00115     std::ofstream d_ems_file;
00116     std::shared_ptr<SbasL1DumpSession> d_dump_session;
00117
00118     size_t d_block_size;                 //!< number of samples which are processed during one
    invocation of the algorithms
00119     std::vector<double> d_sample_buf;    //!< input buffer holding the samples to be processed in one
    block
00120     std::vector<double> d_sample_stamps;  //!< absolute reception timestamp [s] of each sample in
    d_sample_buf, same indexing
00121
00122     // first element: absolute reception timestamp [s] of the message's first bit
00123     typedef std::pair<double, std::vector<int32_t>> msg_candiate_int_t;
00124     typedef std::pair<double, std::vector<uint8_t>> msg_candiate_char_t;
00125
00126     // helper class for sample alignment
00127     class Sample_Aligner
00128     {
00129     public:
00130         Sample_Aligner();
00131         void reset();
00132         /*
00133          * samples length must be a multiple of two
00134          * for block operation
00135          */
00136         bool get_symbols(
00137             const std::vector<double> &samples,
00138             const std::vector<double> &sample_stamps,
00139             std::vector<double> &symbols,
00140             std::vector<double> &symbol_stamps);
00141
00142     private:
00143         int32_t d_n_smpls_in_history{3};
00144         double d_iir_par{0.05};
00145         double d_corr_paired{};
00146         double d_corr_shifted{};
00147         bool d_aligned{};
00148         double d_past_sample{};
00149         double d_past_sample_stamp{};
00150         bool d_has_past_sample{};
00151     } d_sample_aligner;
00152
```

```
00153     // helper class for symbol alignment and Viterbi decoding
00154     class Symbol_Aligner_And_Decoder
00155     {
00156     public:
00157         Symbol_Aligner_And_Decoder();
00158         void reset();
00159         bool get_bits(
00160             const std::vector<double> &symbols,
00161             const std::vector<double> &symbol_stamps,
00162             std::vector<int32_t> &bits,
00163             std::vector<double> &bit_stamps);
00164
00165     private:
00166         int32_t d_KK{7};
00167         std::shared_ptr<Viterbi_Decoder_Sbas> d_vd1;
00168         std::shared_ptr<Viterbi_Decoder_Sbas> d_vd2;
00169         double d_past_symbol{0};
00170         double d_past_symbol_stamp{};
00171         bool d_has_past_symbol{};
00172         std::deque<double> d_pending_bit_stamps_vd1;
00173         std::deque<double> d_pending_bit_stamps_vd2;
00174     } d_symbol_aligner_and_decoder;
00175
00176
00177     // helper class for detecting the preamble and collect the corresponding message candidates
00178     class Frame_Detector
00179     {
00180     public:
00181         void reset();
00182         // bit_stamps[i] must hold the absolute reception timestamp [s] of bits[i]
00183         void get_frame_candidates(const std::vector<int32_t> &bits, const std::vector<double>
      &bit_stamps, std::vector<std::pair<double, std::vector<int32_t>> &msg_candidates);
00184
00185     private:
00186         std::deque<int32_t> d_buffer;
00187         std::deque<double> d_bit_stamps;  //!< absolute reception timestamp [s] of each bit in
      d_buffer, same indexing
00188     } d_frame_detector;
00189
00190
00191     // helper class for checking the CRC of the message candidates
00192     class Crc_Verifier
00193     {
00194     public:
00195         void reset();
00196         void get_valid_frames(const std::vector<msg_candiate_int_t> &msg_candidates,
      std::vector<msg_candiate_char_t> &valid_msgs);
00197
00198     private:
00199         typedef boost::crc_optimal<24, 0x1864CFBU, 0x0, 0x0, false, false> crc_24_q_type;
00200         crc_24_q_type d_checksum_agent;
00201         void zerropad_front_and_convert_to_bytes(const std::vector<int32_t> &msg_candidate,
      std::vector<uint8_t> &bytes);
00202         void zerropad_back_and_convert_to_bytes(const std::vector<int32_t> &msg_candidate,
      std::vector<uint8_t> &bytes);
00203     } d_crc_verifier;
00204 };
00205
00206
00207 /** \} */
00208 /** \} */
00209 #endif  // GNSS_SDR_SBAS_L1_TELEMETRY_DECODER_GS_H
```

13.451 telemetry_impl_interface.h File Reference

Base class for telemetry decoder GNU Radio blocks.

```
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
#include <gnuradio/block.h>
#include <gnuradio/io_signature.h>
#include <cstdint>
#include <fstream>
#include <memory>
#include <string>
#include <utility>
```

Classes

- class telemetry_impl_interface

  Common base class for telemetry decoder GNU Radio implementations.

Typedefs

- using telemetry_impl_interface_sptr = gnss_shared_ptr<telemetry_impl_interface>

13.451.1 Detailed Description

Base class for telemetry decoder GNU Radio blocks.

Author

Carles Fernandez-Prades, 2025 cfernandez@cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file telemetry_impl_interface.h.

13.452 telemetry_impl_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file telemetry_impl_interface.h
00003  * \brief Base class for telemetry decoder GNU Radio blocks.
00004  * \author Carles Fernandez-Prades, 2025 cfernandez@cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_TELEMETRY_IMPL_INTERFACE_H
00018 #define GNSS_SDR_TELEMETRY_IMPL_INTERFACE_H
00019
00020 #include "gnss_block_interface.h"
00021 #include "gnss_satellite.h"
00022 #include <gnuradio/block.h>
00023 #include <gnuradio/io_signature.h>
00024 #include <cstdint>
00025 #include <fstream>
00026 #include <memory>
00027 #include <string>
00028 #include <utility>
00029
00030 class Tlm_Conf;
00031 class Tlm_CRC_Stats;
00032
00033 /** \addtogroup Telemetry_Decoder
00034  * \{
00035  */
00036 /** \addtogroup Telemetry_Decoder_gnuradio_blocks telemetry_decoder_gr_blocks
00037  * \{
00038  */
00039
00040 class telemetry_impl_interface;
00041 using telemetry_impl_interface_sptr = gnss_shared_ptr<telemetry_impl_interface>;
00042
00043 /*!
00044  * \brief Common base class for telemetry decoder GNU Radio implementations.
00045  */
00046 class telemetry_impl_interface : public gr::block
00047 {
00048 public:
00049     telemetry_impl_interface(const std::string& name,
00050         gr::io_signature::sptr input_signature,
00051         gr::io_signature::sptr output_signature)
00052         : gr::block(name,
00053               std::move(input_signature),
```

```
00054                 std::move(output_signature)) {}
00055
00056     ~telemetry_impl_interface() override = default;
00057
00058     virtual void set_satellite(const Gnss_Satellite& satellite) = 0;
00059     virtual void set_channel(int channel) = 0;
00060     virtual void reset() = 0;
00061
00062 protected:
00063     void configure_basic_outputs();
00064
00065     void configure_dump_file(int32_t channel,
00066         bool enable_dump,
00067         std::string& dump_filename,
00068         std::ofstream& dump_file) const;
00069
00070     void configure_crc_stats_channel(int32_t channel,
00071         bool& dump_crc_stats,
00072         std::unique_ptr<Tlm_CRC_Stats>& crc_stats) const;
00073 };
00074
00075 /** \} */
00076 /** \} */
00077
00078 #endif  // GNSS_SDR_TELEMETRY_IMPL_INTERFACE_H
```

13.453 galileo_tow_utils.h File Reference

Small helpers for Galileo TOW arithmetic in telemetry decoder blocks.

```
#include "Galileo_E5a.h"
#include "Galileo_INAV.h"
#include "tow_utils.h"
#include <cstdint>
```

Functions

- int64_t galileo_tow::inav_current_symbol_delay_ms (uint32_t required_symbols, uint32_t code_period_ms)
- int64_t galileo_tow::fnav_current_symbol_delay_ms (uint32_t required_symbols)
- uint32_t galileo_tow::add_ms (uint32_t tow_ms, int64_t delta_ms)
- int64_t galileo_tow::floor_div (int64_t numerator, int64_t denominator)
- uint32_t galileo_tow::floor_to_second_ms (uint32_t tow_ms)
- int64_t galileo_tow::sample_counter_delta (uint64_t current, uint64_t reference)
- uint32_t galileo_tow::seconds_to_ms (int32_t tow_s)
- bool galileo_tow::week_after_delta (uint32_t week, uint32_t tow_ms, int64_t delta_ms, uint32_t &result_↩ week)
- uint32_t galileo_tow::wrap_ms (int64_t tow_ms)

Variables

- constexpr uint32_t galileo_tow::WEEK_MS

13.453.1 Detailed Description

Small helpers for Galileo TOW arithmetic in telemetry decoder blocks.

Author

Carles Fernandez, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_tow_utils.h.

13.453.2 Function Documentation

13.453.2.1 add_ms()

```
uint32_t gnss_tow::add_ms (
            uint32_t tow_ms,
            int64_t delta_ms)  [inline]
```

Definition at line 75 of file tow_utils.h.

13.453.2.2 floor_div()

```
int64_t gnss_tow::floor_div (
            int64_t numerator,
            int64_t denominator)  [inline]
```

Definition at line 99 of file tow_utils.h.

13.453.2.3 floor_to_second_ms()

```
uint32_t gnss_tow::floor_to_second_ms (
            uint32_t tow_ms)  [inline]
```

Definition at line 81 of file tow_utils.h.

13.453.2.4 fnav_current_symbol_delay_ms()

```
int64_t galileo_tow::fnav_current_symbol_delay_ms (
            uint32_t required_symbols)  [inline]
```

Definition at line 44 of file galileo_tow_utils.h.

13.453.2.5 inav_current_symbol_delay_ms()

```
int64_t galileo_tow::inav_current_symbol_delay_ms (
            uint32_t required_symbols,
            uint32_t code_period_ms)  [inline]
```

Definition at line 38 of file galileo_tow_utils.h.

13.453.2.6 sample_counter_delta()

```
int64_t gnss_tow::sample_counter_delta (
            uint64_t current,
            uint64_t reference)  [inline]
```

Definition at line 132 of file tow_utils.h.

13.453.2.7 seconds_to_ms()

```
uint32_t gnss_tow::seconds_to_ms (
            int32_t tow_s)  [inline]
```

Definition at line 69 of file tow_utils.h.

13.453.2.8 week_after_delta()

```
bool gnss_tow::week_after_delta (
            uint32_t week,
            uint32_t tow_ms,
            int64_t delta_ms,
            uint32_t & result_week)  [inline]
```

Definition at line 111 of file tow_utils.h.

13.453.2.9 wrap_ms()

```
uint32_t gnss_tow::wrap_ms (
            int64_t tow_ms)  [inline]
```

Definition at line 35 of file tow_utils.h.

13.453.3 Variable Documentation

13.453.3.1 WEEK_MS

```
uint32_t gnss_tow::WEEK_MS  [constexpr]
```

Definition at line 31 of file tow_utils.h.

13.454 galileo_tow_utils.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_tow_utils.h
00003  * \brief Small helpers for Galileo TOW arithmetic in telemetry decoder blocks
00004  * \author Carles Fernandez, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GALILEO_TOW_UTILS_H
00018 #define GNSS_SDR_GALILEO_TOW_UTILS_H
00019
00020 #include "Galileo_E5a.h"   // for GALILEO_E5A_CODE_PERIOD_MS, GALILEO_FNAV_CODES_PER_SYMBOL
00021 #include "Galileo_INAV.h"  // for GALILEO_INAV_PAGE_PART_MS
00022 #include "tow_utils.h"     // for the system-agnostic TOW arithmetic helpers
00023 #include <cstdint>
00024
00025 namespace galileo_tow
00026 {
00027 // Generic TOW arithmetic, shared with the other telemetry decoders
00028 using gnss_tow::add_ms;
00029 using gnss_tow::floor_div;
00030 using gnss_tow::floor_to_second_ms;
00031 using gnss_tow::sample_counter_delta;
00032 using gnss_tow::seconds_to_ms;
00033 using gnss_tow::week_after_delta;
00034 using gnss_tow::WEEK_MS;
00035 using gnss_tow::wrap_ms;
00036
00037
00038 inline int64_t inav_current_symbol_delay_ms(uint32_t required_symbols, uint32_t code_period_ms)
00039 {
00040     return static_cast<int64_t>(GALILEO_INAV_PAGE_PART_MS) + (static_cast<int64_t>(required_symbols) +
      1LL) * static_cast<int64_t>(code_period_ms);
00041 }
00042
00043
00044 inline int64_t fnav_current_symbol_delay_ms(uint32_t required_symbols)
00045 {
00046     return (static_cast<int64_t>(required_symbols) + 1LL) *
      static_cast<int64_t>(GALILEO_FNAV_CODES_PER_SYMBOL) *
      static_cast<int64_t>(GALILEO_E5A_CODE_PERIOD_MS);
00047 }
00048
00049 }  // namespace galileo_tow
00050
00051 #endif  // GNSS_SDR_GALILEO_TOW_UTILS_H
```

13.455 bits.h File Reference

Utilities for bit manipulation of the libswiftnav library.

```
#include "swift_common.h"
```

Functions

- uint8_t **parity** (uint32_t x)
- uint32_t **getbitu** (const uint8_t *buff, uint32_t pos, uint8_t len)
- int32_t **getbits** (const uint8_t *buff, uint32_t pos, uint8_t len)
- void **setbitu** (uint8_t *buff, uint32_t pos, uint32_t len, uint32_t data)
- void **setbits** (uint8_t *buff, uint32_t pos, uint32_t len, int32_t data)
- void **bitcopy** (void *dst, uint32_t dst_index, const void *src, uint32_t src_index, uint32_t count)
- void **bitshl** (void *buf, uint32_t size, uint32_t shift)
- uint8_t **count_bits_u64** (uint64_t v, uint8_t bv)
- uint8_t **count_bits_u32** (uint32_t v, uint8_t bv)
- uint8_t **count_bits_u16** (uint16_t v, uint8_t bv)
- uint8_t **count_bits_u8** (uint8_t v, uint8_t bv)

13.455.1 Detailed Description

Utilities for bit manipulation of the libswiftnav library.

Author

Fergus Noble fergus@swift-nav.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
This file was originally borrowed from libswiftnav https://github.com/swift-nav/libswiftnav, a portable C library implementing GNSS related functions and algorithms, and then modified by J. Arribas and C. Fernandez
Copyright (C) 2013, 2016 Swift Navigation Inc. Contact: Fergus Noble fergus@swift-nav.com
SPDX-License-Identifier: LGPL-3.0-only
Definition in file bits.h.

13.456 bits.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file bits.h
00003  * \brief Utilities for bit manipulation of the libswiftnav library
00004  * \author Fergus Noble <fergus@swift-nav.com>
00005  *
00006  * ----------------------------------------------------------------------------
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * This file was originally borrowed from libswiftnav
00011  * <https://github.com/swift-nav/libswiftnav>,
00012  * a portable C library implementing GNSS related functions and algorithms,
00013  * and then modified by J. Arribas and C. Fernandez
00014  *
00015  * Copyright (C) 2013, 2016 Swift Navigation Inc.
00016  * Contact: Fergus Noble <fergus@swift-nav.com>
00017  *
00018  * SPDX-License-Identifier: LGPL-3.0-only
00019  *
00020  */
00021
00022 #ifndef GNSS_SDR_BITS_H
00023 #define GNSS_SDR_BITS_H
00024
00025 #include "swift_common.h"
00026
00027 /** \addtogroup Telemetry_Decoder
00028  * \{ */
00029 /** \addtogroup Telemetry_Decoder_libswiftcnav telemetry_decoder_libswiftcnav
00030  * Utilities for CNAV message decoding by Swift Navigation Inc.
00031  * \{ */
00032
00033
```

```
00034 uint8_t parity(uint32_t x);
00035 uint32_t getbitu(const uint8_t *buff, uint32_t pos, uint8_t len);
00036 int32_t getbits(const uint8_t *buff, uint32_t pos, uint8_t len);
00037 void setbitu(uint8_t *buff, uint32_t pos, uint32_t len, uint32_t data);
00038 void setbits(uint8_t *buff, uint32_t pos, uint32_t len, int32_t data);
00039 void bitcopy(void *dst, uint32_t dst_index,
00040     const void *src, uint32_t src_index, uint32_t count);
00041 void bitshl(void *buf, uint32_t size, uint32_t shift);
00042 uint8_t count_bits_u64(uint64_t v, uint8_t bv);
00043 uint8_t count_bits_u32(uint32_t v, uint8_t bv);
00044 uint8_t count_bits_u16(uint16_t v, uint8_t bv);
00045 uint8_t count_bits_u8(uint8_t v, uint8_t bv);
00046
00047
00048 /** \} */
00049 /** \} */
00050 #endif /* GNSS_SDR_BITS_H_ */
```

13.457 cnav_msg.h File Reference

Utilities for CNAV message manipulation of the libswiftnav library.

```
#include "fec.h"
#include "swift_common.h"
#include <limits.h>
#include <stdbool.h>
#include <stdint.h>
#include <stdlib.h>
```

Classes

- struct cnav_msg_t
- struct cnav_v27_part_t
- struct cnav_msg_decoder_t

Macros

- #define GPS_L2_V27_HISTORY_LENGTH_BITS 64
- #define GPS_L2C_V27_INIT_BITS (32)
- #define GPS_L2C_V27_DECODE_BITS (32)
- #define GPS_L2C_V27_DELAY_BITS (32)

Functions

- const v27_poly_t * **cnav_msg_decoder_get_poly** (void)
- void **cnav_msg_decoder_init** (cnav_msg_decoder_t *dec)
- bool **cnav_msg_decoder_add_symbol** (cnav_msg_decoder_t *dec, unsigned char symbol, cnav_msg_t *msg, uint32_t *delay)

13.457.1 Detailed Description

Utilities for CNAV message manipulation of the libswiftnav library.

Author

Valeri Atamaniouk valeri@swift-nav.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
This file was originally borrowed from libswiftnav https://github.com/swift-nav/libswiftnav, a portable C library implementing GNSS related functions and algorithms, and then modified by J. Arribas and C. Fernandez
Copyright (C) 2016 Swift Navigation Inc. Contact: Valeri Atamaniouk valeri@swift-nav.com
SPDX-License-Identifier: LGPL-3.0-only
Definition in file cnav_msg.h.

13.458 cnav_msg.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cnav_msg.h
00003  * \brief Utilities for CNAV message manipulation of the libswiftnav library
00004  * \author Valeri Atamaniouk <valeri@swift-nav.com>
00005  *
00006  * ----------------------------------------------------------------------------
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * This file was originally borrowed from libswiftnav
00011  * <https://github.com/swift-nav/libswiftnav>,
00012  * a portable C library implementing GNSS related functions and algorithms,
00013  * and then modified by J. Arribas and C. Fernandez
00014  *
00015  * Copyright (C) 2016 Swift Navigation Inc.
00016  * Contact: Valeri Atamaniouk <valeri@swift-nav.com>
00017  *
00018  * SPDX-License-Identifier: LGPL-3.0-only
00019  *
00020  */
00021
00022
00023 #ifndef GNSS_SDR_CNAV_MSG_H
00024 #define GNSS_SDR_CNAV_MSG_H
00025
00026 #include "fec.h"
00027 #include "swift_common.h"
00028 #include <limits.h>
00029 #include <stdbool.h>
00030 #include <stdint.h>
00031 #include <stdlib.h>
00032
00033 /** \addtogroup Telemetry_Decoder
00034  * \{ */
00035 /** \addtogroup Telemetry_Decoder_libswiftcnav
00036  * \{ */
00037
00038
00039 /** Size of the Viterbi decoder history. */
00040 #define GPS_L2_V27_HISTORY_LENGTH_BITS 64
00041 /** Bits to accumulate before decoding starts. */
00042 #define GPS_L2C_V27_INIT_BITS (32)
00043 /** Bits to decode at a time. */
00044 #define GPS_L2C_V27_DECODE_BITS (32)
00045 /** Bits in decoder tail. We ignore them. */
00046 #define GPS_L2C_V27_DELAY_BITS (32)
00047 /**
00048  * GPS CNAV message container.
00049  *
00050  * @sa cnav_msg_decoder_add_symbol
00051  */
00052 typedef struct
00053 {
00054     uint8_t prn;                                                    /**< SV PRN. 0..31 */
00055     uint8_t msg_id;                                                 /**< Message id. 0..31 */
00056     uint32_t tow;                                                   /**< GPS ToW in 6-second units.
      Multiply to 6 to get seconds. */
00057     bool alert;                                                     /**< CNAV message alert flag */
00058     uint8_t raw_msg[GPS_L2C_V27_DECODE_BITS + GPS_L2C_V27_DELAY_BITS]; /**< RAW MSG for GNSS-SDR */
00059 } cnav_msg_t;
00060
00061 /**
00062  * GPS CNAV decoder component.
00063  * This component controls symbol decoding string.
00064  *
00065  * @sa cnav_msg_decoder_t
00066  */
00067 typedef struct
00068 {
00069     v27_t dec; /**< Viterbi block decoder object */
00070     v27_decision_t decisions[GPS_L2_V27_HISTORY_LENGTH_BITS];
00071     /**< Decision graph */
00072     unsigned char symbols[(GPS_L2C_V27_INIT_BITS + GPS_L2C_V27_DECODE_BITS) * 2];
00073     /**< Symbol buffer */
00074     size_t n_symbols; /**< Count of symbols in the symbol buffer */
00075     unsigned char decoded[GPS_L2C_V27_DECODE_BITS + GPS_L2C_V27_DELAY_BITS];
00076     /**< Decode buffer */
00077     size_t n_decoded;   /**< Number of bits in the decode buffer */
00078     bool preamble_seen; /**< When true, the decode buffer is aligned on
00079                          *   preamble. */
00080     bool invert;        /**< When true, indicates the bits are inverted */
00081     bool message_lock;  /**< When true, indicates the message boundary
00082                          *   is found. */
```

```
00083     bool crc_ok;        /**< Flag that the last message had good CRC */
00084     size_t n_crc_fail;  /**< Counter for CRC failures */
00085     bool init;          /**< Initial state flag. When true, initial bits
00086                          *   do not produce output. */
00087 } cnav_v27_part_t;
00088
00089 /**
00090  * GPS CNAV message lock and decoder object.
00091  *
00092  * Decoder uses two Viterbi decoder objects to ensure the lock is acquired when
00093  * the input symbol phase is not known.
00094  */
00095 typedef struct
00096 {
00097     cnav_v27_part_t part1; /**< Decoder for odd symbol pairs */
00098     cnav_v27_part_t part2; /**< Decoder for even symbol pairs */
00099 } cnav_msg_decoder_t;
00100
00101 const v27_poly_t *cnav_msg_decoder_get_poly(void);
00102 void cnav_msg_decoder_init(cnav_msg_decoder_t *dec);
00103 bool cnav_msg_decoder_add_symbol(cnav_msg_decoder_t *dec,
00104     unsigned char symbol,
00105     cnav_msg_t *msg,
00106     uint32_t *delay);
00107
00108 /** \} */
00109 /** \} */
00110 #endif /* GNSS_SDR_CNAV_MSG_H_ */
```

13.459 edc.h File Reference

Utilities for CRC computation of the libswiftnav library.
`#include "swift_common.h"`

Functions

- uint32_t **crc24q** (const uint8_t *buf, uint32_t len, uint32_t crc)
- uint32_t **crc24q_bits** (uint32_t crc, const uint8_t *buf, uint32_t n_bits, bool invert)

13.459.1 Detailed Description

Utilities for CRC computation of the libswiftnav library.

Author

Fergus Noble fergus@swift-nav.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
This file was originally borrowed from libswiftnav https://github.com/swift-nav/libswiftnav, a portable C library implementing GNSS related functions and algorithms, and then modified by J. Arribas and C. Fernandez
Copyright (C) 2010 Swift Navigation Inc. Contact: Fergus Noble fergus@swift-nav.com
SPDX-License-Identifier: LGPL-3.0-only
Definition in file edc.h.

13.460 edc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file edc.h
00003  * \brief Utilities for CRC computation of the libswiftnav library
00004  * \author Fergus Noble <fergus@swift-nav.com>
00005  *
00006  * ----------------------------------------------------------------------------
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * This file was originally borrowed from libswiftnav
00011  * <https://github.com/swift-nav/libswiftnav>,
00012  * a portable C library implementing GNSS related functions and algorithms,
```

```
00013  * and then modified by J. Arribas and C. Fernandez
00014  *
00015  * Copyright (C) 2010 Swift Navigation Inc.
00016  * Contact: Fergus Noble <fergus@swift-nav.com>
00017  *
00018  * SPDX-License-Identifier: LGPL-3.0-only
00019  *
00020  */
00021
00022
00023 #ifndef GNSS_SDR_EDC_H
00024 #define GNSS_SDR_EDC_H
00025
00026 #include "swift_common.h"
00027
00028 /** \addtogroup Telemetry_Decoder
00029  * \{ */
00030 /** \addtogroup Telemetry_Decoder_libswiftcnav
00031  * \{ */
00032
00033
00034 uint32_t crc24q(const uint8_t *buf, uint32_t len, uint32_t crc);
00035 uint32_t crc24q_bits(uint32_t crc, const uint8_t *buf, uint32_t n_bits, bool invert);
00036
00037
00038 /** \} */
00039 /** \} */
00040 #endif /* GNSS_SDR_EDC_H */
```

13.461 fec.h File Reference

Utilities for the convolutional encoder of the libswiftnav library.

Classes

- struct v27_poly_t
- struct v27_decision_t
- struct v27_t

Macros

- #define V27POLYA 0x4f
- #define V27POLYB 0x6d

Functions

- void **v27_poly_init** (v27_poly_t *poly, const signed char polynomial[2])
- void **v27_init** (v27_t *v, v27_decision_t *decisions, unsigned int decisions_count, const v27_poly_t *poly, unsigned char initial_state)
- void **v27_update** (v27_t *v, const unsigned char *syms, int nbits)
- void **v27_chainback_fixed** (v27_t *v, unsigned char *data, unsigned int nbits, unsigned char final_state)
- void **v27_chainback_likely** (v27_t *v, unsigned char *data, unsigned int nbits)

13.461.1 Detailed Description

Utilities for the convolutional encoder of the libswiftnav library.

Author

Phil Karn, KA9Q

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
This file was originally borrowed from libswiftnav https://github.com/swift-nav/libswiftnav, a portable C library implementing GNSS related functions and algorithms, and then modified by J. Arribas and C. Fernandez
Copyright (C) 2004, Phil Karn, KA9Q
SPDX-License-Identifier: LGPL-3.0-only
Definition in file fec.h.

13.461.2 Macro Definition Documentation

13.461.2.1 V27POLYA

```
#define V27POLYA 0x4f
```

Definition at line 29 of file fec.h.

13.461.2.2 V27POLYB

```
#define V27POLYB 0x6d
```

Definition at line 30 of file fec.h.

13.462 fec.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fec.h
00003  * \brief Utilities for the convolutional encoder of the libswiftnav library
00004  * \author Phil Karn, KA9Q
00005  *
00006  * -------------------------------------------------------------------------
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * This file was originally borrowed from libswiftnav
00011  * <https://github.com/swift-nav/libswiftnav>,
00012  * a portable C library implementing GNSS related functions and algorithms,
00013  * and then modified by J. Arribas and C. Fernandez
00014  *
00015  * Copyright (C) 2004, Phil Karn, KA9Q
00016  *
00017  * SPDX-License-Identifier: LGPL-3.0-only
00018  *
00019  */
00020
00021
00022 #ifndef GNSS_SDR_FEC_H
00023 #define GNSS_SDR_FEC_H
00024
00025 /* r=1/2 k=7 convolutional encoder polynomials
00026  * The NASA-DSN convention is to use V27POLYA inverted, then V27POLYB
00027  * The CCSDS/NASA-GSFC convention is to use V27POLYB, then V27POLYA inverted
00028  */
00029 #define V27POLYA 0x4f
00030 #define V27POLYB 0x6d
00031
00032 typedef struct
00033 {
00034     unsigned char c0[32];
00035     unsigned char c1[32];
00036 } v27_poly_t;
00037
00038 typedef struct
00039 {
00040     unsigned int w[2];
00041 } v27_decision_t;
00042
00043 /* State info for instance of r=1/2 k=7 Viterbi decoder
00044  */
00045 typedef struct
00046 {
00047     unsigned int metrics1[64]; /* Path metric buffer 1 */
00048     unsigned int metrics2[64]; /* Path metric buffer 2 */
00049     /* Pointers to path metrics, swapped on every bit */
00050     unsigned int *old_metrics, *new_metrics;
00051     const v27_poly_t *poly;       /* Polynomial to use */
00052     v27_decision_t *decisions;    /* Beginning of decisions for block */
00053     unsigned int decisions_index; /* Index of current decision */
00054     unsigned int decisions_count; /* Number of decisions in history */
00055 } v27_t;
00056
00057 void v27_poly_init(v27_poly_t *poly, const signed char polynomial[2]);
00058
00059 void v27_init(v27_t *v, v27_decision_t *decisions, unsigned int decisions_count,
00060     const v27_poly_t *poly, unsigned char initial_state);
00061 void v27_update(v27_t *v, const unsigned char *syms, int nbits);
00062 void v27_chainback_fixed(v27_t *v, unsigned char *data, unsigned int nbits,
00063     unsigned char final_state);
00064 void v27_chainback_likely(v27_t *v, unsigned char *data, unsigned int nbits);
00065
```

```
00066 #endif
```

13.463 swift_common.h File Reference

Common definitions used throughout the libswiftnav library.

```
#include <inttypes.h>
#include <stdbool.h>
#include <stdint.h>
```

Macros

- #define ABS(x)
- #define MIN(x, y)
- #define MAX(x, y)
- #define CLAMP_DIFF(a, b)

13.463.1 Detailed Description

Common definitions used throughout the libswiftnav library.

Author

Henry Hallam henry@swift-nav.com Fergus Noble fergus@swift-nav.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
This file was originally borrowed from libswiftnav https://github.com/swift-nav/libswiftnav, a portable C library implementing GNSS related functions and algorithms, and then modified by J. Arribas and C. Fernandez
Copyright (C) 2012 Swift Navigation Inc. Contact: Henry Hallam henry@swift-nav.com Fergus Noble fergus@swift-nav.com
SPDX-License-Identifier: LGPL-3.0-only
Definition in file swift_common.h.

13.464 swift_common.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file swift_common.h
00003  * \brief Common definitions used throughout the libswiftnav library
00004  * \author Henry Hallam <henry@swift-nav.com>
00005  *         Fergus Noble <fergus@swift-nav.com>
00006  *
00007  * ----------------------------------------------------------------------------
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * This file was originally borrowed from libswiftnav
00012  * <https://github.com/swift-nav/libswiftnav>,
00013  * a portable C library implementing GNSS related functions and algorithms,
00014  * and then modified by J. Arribas and C. Fernandez
00015  *
00016  * Copyright (C) 2012 Swift Navigation Inc.
00017  * Contact: Henry Hallam <henry@swift-nav.com>
00018  *          Fergus Noble <fergus@swift-nav.com>
00019  *
00020  * SPDX-License-Identifier: LGPL-3.0-only
00021  *
00022  */
00023
00024
00025 #ifndef GNSS_SDR_SWIFT_COMMON_H
00026 #define GNSS_SDR_SWIFT_COMMON_H
00027
00028 /** \addtogroup Telemetry_Decoder
00029  * \{ */
00030 /** \addtogroup Telemetry_Decoder_libswiftcnav
00031  * \{ */
```

```
00032
00033
00034 #define ABS(x) ((x) < 0 ? -(x) : (x))
00035 #define MIN(x, y) (((x) < (y)) ? (x) : (y))
00036 #define MAX(x, y) (((x) > (y)) ? (x) : (y))
00037 #define CLAMP_DIFF(a, b) (MAX((a), (b)) - (b))
00038
00039 #include <inttypes.h>
00040 #include <stdbool.h>
00041 #include <stdint.h>
00042
00043
00044 /** \} */
00045 /** \} */
00046 #endif /* GNSS_SDR_SWIFT_COMMON_H */
```

13.465 tlm_conf.h File Reference

Class that contains all the configuration parameters for generic telemetry decoder block.

```
#include "configuration_interface.h"
#include <string>
```

Classes

- class Tlm_Conf

13.465.1 Detailed Description

Class that contains all the configuration parameters for generic telemetry decoder block.

Author

Carles Fernandez, 2020. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tlm_conf.h.

13.466 tlm_conf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tlm_conf.h
00003  * \brief Class that contains all the configuration parameters for generic
00004  * telemetry decoder block.
00005  * \author Carles Fernandez, 2020. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_TLM_CONF_H
00019 #define GNSS_SDR_TLM_CONF_H
00020
00021 #include "configuration_interface.h"
00022 #include <string>
00023
00024 /** \addtogroup Telemetry_Decoder
00025  * \{ */
00026 /** \addtogroup Telemetry_Decoder_libs
00027  * \{ */
00028
00029
00030 class Tlm_Conf
00031 {
00032 public:
```

```
00033     Tlm_Conf() = default;
00034
00035     void SetFromConfiguration(const ConfigurationInterface *configuration, const std::string &role);
00036
00037     std::string dump_filename;
00038     std::string dump_crc_stats_filename;
00039     bool dump{false};
00040     bool dump_mat{false};
00041     bool remove_dat{false};
00042     bool enable_reed_solomon{false};  // for INAV message in Galileo E1B
00043     bool dump_crc_stats{false};       // telemetry CRC statistics
00044     bool enable_navdata_monitor{false};
00045     bool there_are_e1_channels{false};
00046     bool there_are_e6_channels{false};
00047     bool use_ced{false};
00048     bool tow_to_trk{false};
00049 };
00050
00051
00052 /** \} */
00053 /** \} */
00054 #endif  // GNSS_SDR_TLM_CONF_H
```

13.467 tlm_crc_stats.h File Reference

Class that computes the telemetry CRC statistics.

```
#include <cstdint>
#include <fstream>
#include <string>
```

Classes

- class Tlm_CRC_Stats

  Class that computes the telemetry CRC statistics.

13.467.1 Detailed Description

Class that computes the telemetry CRC statistics.

Author

Marc Majoral, 2021. mmajoral(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tlm_crc_stats.h.

13.468 tlm_crc_stats.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tlm_crc_stats.h
00003  * \brief Class that computes the telemetry CRC statistics
00004  * \author Marc Majoral, 2021. mmajoral(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_CRC_STATS_H
00018 #define GNSS_SDR_CRC_STATS_H
00019
00020 #include <cstdint>
00021 #include <fstream>  // for std::ofstream
```

```
00022 #include <string>   // for std::string
00023
00024 /** \addtogroup Telemetry_Decoder
00025  * \{ */
00026 /** \addtogroup Telemetry_Decoder_libs telemetry_decoder_libs
00027  * \{ */
00028
00029 /*!
00030  * \brief Class that computes the telemetry CRC statistics
00031  */
00032 class Tlm_CRC_Stats
00033 {
00034 public:
00035     Tlm_CRC_Stats() = default;
00036
00037     ~Tlm_CRC_Stats();
00038
00039     /*!
00040      * \brief Initialize the telemetry CRC statistics
00041      */
00042     void initialize(std::string dump_crc_stats_filename);
00043
00044     /*!
00045      * \brief Initialize the channel number and output file
00046      */
00047     bool set_channel(int32_t channel);
00048
00049     /*!
00050      * \brief Update the CRC statistics
00051      */
00052     void update_CRC_stats(bool CRC);
00053
00054 private:
00055     std::ofstream d_dump_file;
00056     std::string d_dump_crc_stats_filename;
00057     uint32_t d_num_crc_ok{0};
00058     uint32_t d_num_crc_not_ok{0};
00059     int32_t d_channel{0};
00060     bool d_enable_crc_stats{false};
00061 };
00062
00063
00064 /** \} */
00065 /** \} */
00066 #endif  // GNSS_SDR_CRC_STATS_H
```

13.469 tlm_utils.h File Reference

Utilities for the telemetry decoder blocks.

`#include <string>`

Functions

- int **save_tlm_matfile** (const std::string &dumpfile)
- bool **tlm_remove_file** (const std::string &file_to_remove)
- void **tlm_cleanup_and_save_files** (std::ofstream &dump_file, const std::string &dump_filename, bool dump, bool dump_mat, bool remove_dat)

13.469.1 Detailed Description

Utilities for the telemetry decoder blocks.

Author

Carles Fernandez, 2020. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tlm_utils.h.

13.470 tlm_utils.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tlm_utils.h
00003  * \brief Utilities for the telemetry decoder blocks.
00004  * \author Carles Fernandez, 2020. cfernandez(at)cttc.es
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_TLM_UTILS_H
00018 #define GNSS_SDR_TLM_UTILS_H
00019
00020 #include <string>
00021
00022 /** \addtogroup Telemetry_Decoder
00023  * \{ */
00024 /** \addtogroup Telemetry_Decoder_libs
00025  * \{ */
00026
00027 int save_tlm_matfile(const std::string &dumpfile);
00028
00029 bool tlm_remove_file(const std::string &file_to_remove);
00030
00031 void tlm_cleanup_and_save_files(std::ofstream &dump_file, const std::string &dump_filename, bool dump,
      bool dump_mat, bool remove_dat);
00032
00033 /** \} */
00034 /** \} */
00035 #endif  // GNSS_SDR_TLM_UTILS_H
```

13.471 tow_utils.h File Reference

Small helpers for TOW arithmetic shared by telemetry decoder blocks.

```
#include <cmath>
#include <cstdint>
#include <limits>
```

Namespaces

- namespace gnss_tow

Functions

- uint32_t gnss_tow::wrap_ms (int64_t tow_ms)
- double gnss_tow::wrap_s (double tow_s)
- uint32_t gnss_tow::wrap_s_to_ms (double tow_s)
- int64_t gnss_tow::to_ms (double tow_s)
- uint32_t gnss_tow::seconds_to_ms (int32_t tow_s)
- uint32_t gnss_tow::add_ms (uint32_t tow_ms, int64_t delta_ms)
- uint32_t gnss_tow::floor_to_second_ms (uint32_t tow_ms)
- uint32_t gnss_tow::circular_error_ms (uint32_t lhs_ms, uint32_t rhs_ms)
- int64_t gnss_tow::floor_div (int64_t numerator, int64_t denominator)
- bool gnss_tow::week_after_delta (uint32_t week, uint32_t tow_ms, int64_t delta_ms, uint32_t &result_week)
- int64_t gnss_tow::sample_counter_delta (uint64_t current, uint64_t reference)

Variables

- constexpr uint32_t gnss_tow::WEEK_MS = 604800000U
- constexpr double gnss_tow::WEEK_S = 604800.0

13.471.1 Detailed Description

Small helpers for TOW arithmetic shared by telemetry decoder blocks.

Author

Carles Fernandez, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file tow_utils.h.

13.472 tow_utils.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tow_utils.h
00003  * \brief Small helpers for TOW arithmetic shared by telemetry decoder blocks
00004  * \author Carles Fernandez, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_TOW_UTILS_H
00018 #define GNSS_SDR_TOW_UTILS_H
00019
00020 #include <cmath>
00021 #include <cstdint>
00022 #include <limits>
00023
00024 /*!
00025  * Helpers for arithmetic on the GPS time of week reported by the telemetry
00026  * decoders, which spans [0, 604800) seconds and rolls over at the week
00027  * boundary.
00028  */
00029 namespace gnss_tow
00030 {
00031 constexpr uint32_t WEEK_MS = 604800000U;
00032 constexpr double WEEK_S = 604800.0;
00033
00034
00035 inline uint32_t wrap_ms(int64_t tow_ms)
00036 {
00037     tow_ms %= static_cast<int64_t>(WEEK_MS);
00038     if (tow_ms < 0)
00039         {
00040             tow_ms += static_cast<int64_t>(WEEK_MS);
00041         }
00042     return static_cast<uint32_t>(tow_ms);
00043 }
00044
00045
00046 inline double wrap_s(double tow_s)
00047 {
00048     double wrapped_tow_s = std::fmod(tow_s, WEEK_S);
00049     if (wrapped_tow_s < 0.0)
00050         {
00051             wrapped_tow_s += WEEK_S;
00052         }
00053     return wrapped_tow_s;
00054 }
00055
00056
00057 inline uint32_t wrap_s_to_ms(double tow_s)
00058 {
00059     return static_cast<uint32_t>(std::round(wrap_s(tow_s) * 1000.0)) % WEEK_MS;
00060 }
00061
00062
00063 inline int64_t to_ms(double tow_s)
00064 {
00065     return static_cast<int64_t>(tow_s * 1000.0);
00066 }
00067
```

```
00068
00069 inline uint32_t seconds_to_ms(int32_t tow_s)
00070 {
00071     return wrap_ms(static_cast<int64_t>(tow_s) * 1000LL);
00072 }
00073
00074
00075 inline uint32_t add_ms(uint32_t tow_ms, int64_t delta_ms)
00076 {
00077     return wrap_ms(static_cast<int64_t>(tow_ms) + delta_ms);
00078 }
00079
00080
00081 inline uint32_t floor_to_second_ms(uint32_t tow_ms)
00082 {
00083     return (tow_ms / 1000U) * 1000U;
00084 }
00085
00086
00087 /*!
00088  * Distance in ms between two TOW values along the shortest direction of the
00089  * week circle.
00090  */
00091 inline uint32_t circular_error_ms(uint32_t lhs_ms, uint32_t rhs_ms)
00092 {
00093     const uint32_t forward_error_ms = lhs_ms >= rhs_ms ? lhs_ms - rhs_ms : lhs_ms + WEEK_MS - rhs_ms;
00094     const uint32_t reverse_error_ms = rhs_ms >= lhs_ms ? rhs_ms - lhs_ms : rhs_ms + WEEK_MS - lhs_ms;
00095     return forward_error_ms < reverse_error_ms ? forward_error_ms : reverse_error_ms;
00096 }
00097
00098
00099 inline int64_t floor_div(int64_t numerator, int64_t denominator)
00100 {
00101     const int64_t quotient = numerator / denominator;
00102     const int64_t remainder = numerator % denominator;
00103     if (remainder != 0 && ((remainder < 0) != (denominator < 0)))
00104         {
00105             return quotient - 1;
00106         }
00107     return quotient;
00108 }
00109
00110
00111 inline bool week_after_delta(uint32_t week, uint32_t tow_ms, int64_t delta_ms, uint32_t& result_week)
00112 {
00113     const int64_t week_delta = floor_div(static_cast<int64_t>(tow_ms) + delta_ms,
    static_cast<int64_t>(WEEK_MS));
00114     if (week_delta < 0)
00115         {
00116             const uint64_t negative_week_delta = static_cast<uint64_t>(-(week_delta + 1LL)) + 1ULL;
00117             if (negative_week_delta > static_cast<uint64_t>(week))
00118                 {
00119                     return false;
00120                 }
00121         }
00122     else if (week_delta > static_cast<int64_t>(std::numeric_limits<uint32_t>::max() - week))
00123         {
00124             return false;
00125         }
00126
00127     result_week = static_cast<uint32_t>(static_cast<int64_t>(week) + week_delta);
00128     return true;
00129 }
00130
00131
00132 inline int64_t sample_counter_delta(uint64_t current, uint64_t reference)
00133 {
00134     const auto max_delta = static_cast<uint64_t>(std::numeric_limits<int64_t>::max());
00135     if (current >= reference)
00136         {
00137             const uint64_t delta = current - reference;
00138             return delta > max_delta ? std::numeric_limits<int64_t>::max() :
    static_cast<int64_t>(delta);
00139         }
00140
00141     const uint64_t delta = reference - current;
00142     return delta > max_delta ? std::numeric_limits<int64_t>::min() + 1 : -static_cast<int64_t>(delta);
00143 }
00144
00145 }  // namespace gnss_tow
00146
00147 #endif  // GNSS_SDR_TOW_UTILS_H
```

13.473 viterbi_decoder.h File Reference

Class that implements a Viterbi decoder.

```
#include <array>
#include <cstdint>
#include <limits>
#include <vector>
```

Classes

- class Viterbi_Decoder

  Class that implements a Viterbi decoder.

13.473.1 Detailed Description

Class that implements a Viterbi decoder.

Author

Carles Fernandez, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file viterbi_decoder.h.

13.474 viterbi_decoder.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file viterbi_decoder.h
00003  * \brief Class that implements a Viterbi decoder
00004  * \author Carles Fernandez, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_VITERBI_DECODER_H
00018 #define GNSS_SDR_VITERBI_DECODER_H
00019
00020 #include <array>
00021 #include <cstdint>
00022 #include <limits>
00023 #include <vector>
00024
00025 /** \addtogroup Telemetry_Decoder
00026  * \{ */
00027 /** \addtogroup Telemetry_Decoder_libs
00028  * Utilities for the decoding of GNSS navigation messages.
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief Class that implements a Viterbi decoder
00034  */
00035 class Viterbi_Decoder
00036 {
00037 public:
00038     /*!
00039      * \brief Constructor of a Viterbi decoder
00040      * \param[in] KK  Constraint length
00041      * \param[in] nn  Coding rate 1/n
00042      * \param[in] LL  Data length
00043      * \param[in] g   Polynomial G1 and G2
00044      */
00045     Viterbi_Decoder(int32_t KK, int32_t nn, int32_t LL, const std::array<int32_t, 2>& g);
```

```
00046
00047     /*!
00048      * \brief Uses the Viterbi algorithm to perform soft-decision decoding
00049      *        of a convolutional code.
00050      * \param[out] output_u_int Hard decisions on the decoded data bits.
00051      * \param[in] input_c Soft input metrics.
00052      */
00053     void decode(std::vector<int32_t>& output_u_int, const std::vector<float>& input_c);
00054
00055     /*!
00056      * \brief Reset internal status
00057      */
00058     void reset();
00059
00060 private:
00061     static float neg_inf() noexcept
00062     {
00063         return -std::numeric_limits<float>::infinity();
00064     }
00065
00066     /*
00067      * Function that creates the transit and output vectors
00068      */
00069     void nsc_transit(std::vector<int32_t>& output_p,
00070         std::vector<int32_t>& trans_p,
00071         int32_t input) const;
00072
00073     /*
00074      *  Computes the branch metric used for decoding.
00075      *  \return (returned float) The metric between the hypothetical symbol and the received vector
00076      *  \param[in] symbol        The hypothetical symbol
00077      *
00078      */
00079     float Gamma(int32_t symbol) const;
00080
00081     /*
00082      * Determines if a symbol has odd (1) or even (0) parity
00083      *    Output parameters:
00084      * \return (returned int): The symbol's parity = 1 for odd and 0 for even
00085      *
00086      * \param[in] symbol  The integer-valued symbol
00087      * \param[in] length  The highest bit position in the symbol
00088      *
00089      * This function is used by nsc_enc_bit()
00090      */
00091     int32_t parity_counter(int32_t symbol, int32_t length) const;
00092
00093     /*
00094      * Convolutionally encodes a single bit using a rate 1/n encoder.
00095      * Takes in one input bit at a time, and produces a n-bit output.
00096      *
00097      * \return (returned int): Computed output
00098      *
00099      * \param[in]  input     The input data bit (i.e. a 0 or 1).
00100      * \param[in]  state_in  The starting state of the encoder (an int from 0 to 2^m-1).
00101      * \param[out] state_out_p[]  An integer containing the final state of the encoder
00102      *                            (i.e. the state after encoding this bit)
00103      *
00104      * This function is used by nsc_transit()
00105      */
00106     int32_t nsc_enc_bit(int32_t* state_out_p,
00107         int32_t input,
00108         int32_t state_in) const;
00109
00110     std::vector<float> d_prev_section{};
00111     std::vector<float> d_next_section{};
00112
00113     std::vector<float> d_rec_array{};
00114     std::vector<float> d_metric_c{};
00115     std::vector<int32_t> d_prev_bit{};
00116     std::vector<int32_t> d_prev_state{};
00117     std::array<int32_t, 2> d_g{};
00118
00119     std::vector<int32_t> d_out0;
00120     std::vector<int32_t> d_out1;
00121     std::vector<int32_t> d_state0;
00122     std::vector<int32_t> d_state1;
00123
00124     int32_t d_KK{};
00125     int32_t d_nn{};
00126     int32_t d_LL{};
00127
00128     int32_t d_mm{};
00129     int32_t d_states{};
00130     int32_t d_number_symbols{};
00131 };
00132
```

```
00133 /** \} */
00134 /** \} */
00135 #endif  // GNSS_SDR_VITERBI_DECODER_H
```

13.475 viterbi_decoder_sbas.h File Reference

Interface of a Viterbi decoder class based on the Iterative Solutions Coded Modulation Library by Matthew C. Valenti.

```
#include <cstddef>
#include <deque>
#include <vector>
```

Classes

- class Viterbi_Decoder_Sbas

  Class that implements a Viterbi decoder.

13.475.1 Detailed Description

Interface of a Viterbi decoder class based on the Iterative Solutions Coded Modulation Library by Matthew C. Valenti.

Author

Daniel Fehr 2013. daniel.co(at)bluewin.ch

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file viterbi_decoder_sbas.h.

13.476 viterbi_decoder_sbas.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file viterbi_decoder_sbas.h
00003  * \brief Interface of a Viterbi decoder class based on the Iterative Solutions
00004  * Coded Modulation Library by Matthew C. Valenti
00005  * \author Daniel Fehr 2013. daniel.co(at)bluewin.ch
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_VITERBI_DECODER_SBAS_H
00019 #define GNSS_SDR_VITERBI_DECODER_SBAS_H
00020
00021 #include <cstddef>  // for size_t
00022 #include <deque>
00023 #include <vector>
00024
00025 /** \addtogroup Telemetry_Decoder
00026  * \{ */
00027 /** \addtogroup Telemetry_Decoder_libs telemetry_decoder_libs
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief Class that implements a Viterbi decoder
00033  */
00034 class Viterbi_Decoder_Sbas
00035 {
00036 public:
00037     Viterbi_Decoder_Sbas(const int g_encoder[], int KK, int nn);
00038
00039     void reset();
00040
```

```
00041     /*!
00042      * \brief Uses the Viterbi algorithm to perform hard-decision decoding of a convolutional code.
00043      *
00044      * \param[in]  input_c[]    The received signal in LLR-form. For BPSK, must be in form r =
      2*a*y/(sigma^2).
00045      * \param[in]  LL           The number of data bits to be decoded (does not include the mm
      zero-tail-bits)
00046      *
00047      * \return  output_u_int[] Hard decisions on the data bits (without the mm zero-tail-bits)
00048      */
00049     float decode_block(const double input_c[], int* output_u_int, int LL);
00050
00051     float decode_continuous(const double sym[], int traceback_depth, int bits[],
00052         int nbits_requested, int& nbits_decoded);
00053
00054 private:
00055     class Prev
00056     {
00057     public:
00058         int num_states;
00059         Prev(int states, int tt);
00060         Prev(const Prev& prev);
00061         Prev& operator=(const Prev& other);
00062         ~Prev();
00063
00064         int get_anchestor_state_of_current_state(int current_state) const;
00065         int get_bit_of_current_state(int current_state) const;
00066         float get_metric_of_current_state(int current_state) const;
00067         int get_t() const;
00068         void set_current_state_as_ancestor_of_next_state(int next_state, int current_state);
00069         void set_decoded_bit_for_next_state(int next_state, int bit);
00070         void set_survivor_branch_metric_of_next_state(int next_state, float metric);
00071
00072     private:
00073         std::vector<float> v_metric;
00074         std::vector<int> state;
00075         std::vector<int> v_bit;
00076         int t;
00077         int refcount;
00078     };
00079
00080     // operations on the trellis (change decoder state)
00081     void init_trellis_state();
00082     int do_acs(const double sym[], int nbits);
00083     int do_traceback(std::size_t traceback_length);
00084     int do_tb_and_decode(int traceback_length, int requested_decoding_length, int state, int
      output_u_int[], float& indicator_metric);
00085
00086     // branch metric function
00087     float gamma(const float rec_array[], int symbol, int nn);
00088
00089     // trellis generation
00090     void nsc_transit(int output_p[], int trans_p[], int input, const int g[], int KK, int nn);
00091     int nsc_enc_bit(int state_out_p[], int input, int state_in, const int g[], int KK, int nn);
00092     int parity_counter(int symbol, int length);
00093
00094     // trellis state
00095     std::deque<Prev> d_trellis_paths;
00096     std::vector<float> d_pm_t;
00097     std::vector<float> d_metric_c;  /* Set of all possible branch metrics */
00098     std::vector<float> d_rec_array; /* Received values for one trellis section */
00099
00100     // trellis definition
00101     std::vector<int> d_out0;
00102     std::vector<int> d_state0;
00103     std::vector<int> d_out1;
00104     std::vector<int> d_state1;
00105
00106     // measures
00107     float d_indicator_metric;
00108
00109     // code properties
00110     int d_KK;
00111     int d_nn;
00112
00113     // derived code properties
00114     int d_mm;
00115     int d_states;
00116     int d_number_symbols;
00117     bool d_trellis_state_is_initialised;
00118 };
00119
00120
00121 /** \} */
00122 /** \} */
00123 #endif  // GNSS_SDR_VITERBI_DECODER_SBAS_H
```

13.477 dll_pll_tracking_adapter.h File Reference

Base class providing shared logic for DLL+PLL VEML tracking adapters.

```
#include "dll_pll_conf.h"
#include "dll_pll_veml_tracking.h"
#include "signal_flag.h"
#include "tracking_interface.h"
#include <cstddef>
#include <string>
```

Classes

- class DllPllTrackingAdapter

  Base class providing shared logic for DLL+PLL tracking loop adapters for GNSS signals.

13.477.1 Detailed Description

Base class providing shared logic for DLL+PLL VEML tracking adapters.

Authors

Carles Fernandez, 2025. carles.fernandez(at)cttc.cat Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.↩ com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file dll_pll_tracking_adapter.h.

13.478 dll_pll_tracking_adapter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_tracking_adapter.h
00003  * \brief Base class providing shared logic for DLL+PLL VEML tracking adapters.
00004  * \authors Carles Fernandez, 2025. carles.fernandez(at)cttc.cat
00005  *          Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.com
00006  *
00007  * ----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * ----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_H
00019 #define GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_H
00020
00021 #include "dll_pll_conf.h"
00022 #include "dll_pll_veml_tracking.h"
00023 #include "signal_flag.h"
00024 #include "tracking_interface.h"
00025 #include <cstddef>
00026 #include <string>
00027
00028 /** \addtogroup Tracking
00029  * Classes for GNSS signal tracking.
00030  * \{ */
00031 /** \addtogroup Tracking_adapters tracking_adapters
00032  * Wrap GNU Radio blocks for GNSS signal tracking with a TrackingInterface
00033  * \{ */
00034
00035 class ConfigurationInterface;
00036
00037 /*!
00038  * \brief Base class providing shared logic for DLL+PLL tracking loop adapters
00039  * for GNSS signals.
00040  */
00041 class DllPllTrackingAdapter : public TrackingInterface
```

```
00042 {
00043 public:
00044     /*!
00045      * \brief Base constructor of a Tracking block adapter
00046      */
00047     explicit DllPllTrackingAdapter(const ConfigurationInterface* configuration,
00048         std::string role,
00049         std::string implementation,
00050         unsigned int in_streams,
00051         unsigned int out_streams,
00052         signal_flag sig_flag);
00053
00054     /*!
00055      * \brief Default destructor of the Tracking block adapter
00056      */
00057     ~DllPllTrackingAdapter() override = default;
00058
00059     /*!
00060      * \brief Get role from the Tracking block adapter
00061      */
00062     inline std::string role() override { return role_; }
00063
00064     /*!
00065      * \brief Get implementation from the Tracking block adapter
00066      */
00067     inline std::string implementation() override { return implementation_; }
00068
00069     /*!
00070      * \brief Get item_size from the Tracking block adapter
00071      */
00072     inline size_t item_size() override
00073     {
00074         return item_size_;
00075     }
00076
00077     /*!
00078      * \brief Connect the Tracking block adapter
00079      */
00080     void connect(gr::top_block_sptr top_block) override;
00081
00082     /*!
00083      * \brief Disconnect the sTracking block adapter
00084      */
00085     void disconnect(gr::top_block_sptr top_block) override;
00086
00087     /*!
00088      * \brief Get left block from the Tracking block adapter
00089      */
00090     gr::basic_block_sptr get_left_block() override;
00091
00092     /*!
00093      * \brief Get right block from the Tracking block adapter
00094      */
00095     gr::basic_block_sptr get_right_block() override;
00096
00097     /*!
00098      * \brief Set tracking channel unique ID
00099      */
00100     void set_channel(unsigned int channel) override;
00101
00102     /*!
00103      * \brief Set acquisition Gnss_Synchro object pointer
00104      * to exchange synchronization data between acquisition and tracking blocks
00105      */
00106     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00107
00108     /*!
00109      * \brief Start the Tracking block
00110      */
00111     void start_tracking() override;
00112
00113     /*!
00114      * \brief Stop the Tracking block
00115      */
00116     void stop_tracking() override;
00117
00118 private:
00119     dll_pll_veml_tracking_sptr tracking_sptr_;
00120     Dll_Pll_Conf trk_params_;
00121     const std::string role_;
00122     const std::string implementation_;
00123     size_t item_size_;
00124 };
00125
00126 /** \} */
00127 /** \} */
00128 #endif  // GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_H
```

13.479 dll_pll_tracking_adapter_fpga.h File Reference

Adapts an FPGA-offloaded DLL/PLL tracking block to a TrackingInterface.

```
#include "dll_pll_conf_fpga.h"
#include "dll_pll_veml_tracking_fpga.h"
#include "signal_flag.h"
#include "tracking_interface.h"
#include <cstddef>
#include <cstdint>
#include <map>
#include <memory>
#include <mutex>
#include <string>
```

Classes

- class DllPllTrackingAdapterFpga

13.479.1 Detailed Description

Adapts an FPGA-offloaded DLL/PLL tracking block to a TrackingInterface.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dll_pll_tracking_adapter_fpga.h.

13.480 dll_pll_tracking_adapter_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_tracking_adapter_fpga.h
00003  * \brief Adapts an FPGA-offloaded DLL/PLL tracking block to a TrackingInterface
00004  *
00005  * -----------------------------------------------------------------------------
00006  *
00007  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00008  * This file is part of GNSS-SDR.
00009  *
00010  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00011  * SPDX-License-Identifier: GPL-3.0-or-later
00012  *
00013  * -----------------------------------------------------------------------------
00014  */
00015
00016
00017 #ifndef GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_FPGA_H
00018 #define GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_FPGA_H
00019
00020 #include "dll_pll_conf_fpga.h"
00021 #include "dll_pll_veml_tracking_fpga.h"
00022 #include "signal_flag.h"
00023 #include "tracking_interface.h"
00024 #include <cstddef>
00025 #include <cstdint>
00026 #include <map>
00027 #include <memory>
00028 #include <mutex>
00029 #include <string>
00030
00031 class ConfigurationInterface;
00032
00033 class DllPllTrackingAdapterFpga : public TrackingInterface
00034 {
00035 public:
00036     DllPllTrackingAdapterFpga(const ConfigurationInterface* configuration,
00037         const std::string& role,
00038         const std::string& implementation,
00039         unsigned int in_streams,
00040         unsigned int out_streams,
00041         signal_flag sig_flag);
00042
00043     ~DllPllTrackingAdapterFpga() override;
00044
```

```
00045     inline std::string role() override { return role_; }
00046     inline std::string implementation() override { return implementation_; }
00047     inline size_t item_size() override { return sizeof(int16_t); }
00048
00049     void connect(gr::top_block_sptr top_block) override;
00050     void disconnect(gr::top_block_sptr top_block) override;
00051     gr::basic_block_sptr get_left_block() override;
00052     gr::basic_block_sptr get_right_block() override;
00053
00054     void start_tracking() override;
00055     void stop_tracking() override;
00056     void set_channel(unsigned int channel) override;
00057     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00058
00059 private:
00060     void configure_fpga_tracking_channel_mapping(const std::string& signal);
00061     void set_signal_channel_base_index_locked();
00062     uint32_t get_num_alternative_devices_locked() const;
00063     void configure_signal_parameters(const ConfigurationInterface* configuration);
00064     void generate_prn_codes();
00065
00066     static const std::map<std::string, std::string> signal_to_device_;
00067     static const std::map<std::string, std::string> signal_to_alternative_device_;
00068     static std::map<std::string, int> channel_counts_;
00069     static std::mutex channel_counts_mtx_;
00070
00071     Dll_Pll_Conf_Fpga trk_params_;
00072     dll_pll_veml_tracking_fpga_sptr tracking_fpga_sc_sptr_;
00073     int32_t* prn_codes_ptr_;
00074     int32_t* data_codes_ptr_;
00075     const signal_flag sig_flag_;
00076     const std::string role_;
00077     const std::string implementation_;
00078     std::string signal_;
00079     std::string device_name_;
00080     uint32_t channel_;
00081     uint32_t signal_base_channel_index_;
00082 };
00083
00084 #endif  // GNSS_SDR_DLL_PLL_TRACKING_ADAPTER_FPGA_H
```

13.481 galileo_e1_tcp_connector_tracking.h File Reference

Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for Galileo E1 to a TrackingInterface.

```
#include "galileo_e1_tcp_connector_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

Classes

- class GalileoE1TcpConnectorTracking

  This class implements a code DLL + carrier PLL tracking loop.

13.481.1 Detailed Description

Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for Galileo E1 to a TrackingInterface.

Author

David Pubill, 2012. dpubill(at)cttc.es Luis Esteve, 2012. luis(at)epsilon-formacion.com Javier Arribas, 2011. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2012-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_e1_tcp_connector_tracking.h.

13.482 galileo_e1_tcp_connector_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e1_tcp_connector_tracking.h
00003  * \brief  Interface of an adapter of a TCP connector block based on code DLL + carrier PLL
00004  * for Galileo E1 to a TrackingInterface
00005  * \author David Pubill, 2012. dpubill(at)cttc.es
00006  *         Luis Esteve, 2012. luis(at)epsilon-formacion.com
00007  *         Javier Arribas, 2011. jarribas(at)cttc.es
00008  *
00009  * Code DLL + carrier PLL according to the algorithms described in:
00010  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00011  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00012  * Approach, Birkhauser, 2007
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00017  * This file is part of GNSS-SDR.
00018  *
00019  * Copyright (C) 2012-2020  (see AUTHORS file for a list of contributors)
00020  * SPDX-License-Identifier: GPL-3.0-or-later
00021  *
00022  * -----------------------------------------------------------------------------
00023  */
00024
00025 #ifndef GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_H
00026 #define GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_H
00027
00028 #include "galileo_e1_tcp_connector_tracking_cc.h"
00029 #include "tracking_interface.h"
00030 #include <string>
00031
00032 /** \addtogroup Tracking
00033  * \{ */
00034 /** \addtogroup Tracking_adapters
00035  * \{ */
00036
00037
00038 class ConfigurationInterface;
00039
00040 /*!
00041  * \brief This class implements a code DLL + carrier PLL tracking loop
00042  */
00043 class GalileoE1TcpConnectorTracking : public TrackingInterface
00044 {
00045 public:
00046     GalileoE1TcpConnectorTracking(
00047         const ConfigurationInterface* configuration,
00048         const std::string& role,
00049         unsigned int in_streams,
00050         unsigned int out_streams);
00051
00052     ~GalileoE1TcpConnectorTracking() = default;
00053
00054     inline std::string role() override
00055     {
00056         return role_;
00057     }
00058
00059     //! Returns "Galileo_E1_TCP_CONNECTOR_Tracking"
00060     inline std::string implementation() override
00061     {
00062         return "Galileo_E1_TCP_CONNECTOR_Tracking";
00063     }
00064
00065     inline size_t item_size() override
00066     {
00067         return item_size_;
00068     }
00069
00070     void connect(gr::top_block_sptr top_block) override;
00071     void disconnect(gr::top_block_sptr top_block) override;
00072     gr::basic_block_sptr get_left_block() override;
00073     gr::basic_block_sptr get_right_block() override;
00074
00075     /*!
00076      * \brief Set tracking channel unique ID
00077      */
00078     void set_channel(unsigned int channel) override;
00079
00080     /*!
00081      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00082      * to efficiently exchange synchronization data between acquisition and
00083      *  tracking blocks
```

```
00084      */
00085     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00086
00087     void start_tracking() override;
00088     /*!
00089      * \brief Stop running tracking
00090      */
00091     void stop_tracking() override;
00092
00093 private:
00094     galileo_e1_tcp_connector_tracking_cc_sptr tracking_sptr_;
00095     std::string role_;
00096     size_t item_size_;
00097     unsigned int channel_;
00098     unsigned int in_streams_;
00099     unsigned int out_streams_;
00100 };
00101
00102
00103 /** \} */
00104 /** \} */
00105 #endif  // GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_H
```

13.483 gps_l1_ca_dll_pll_tracking_gpu.h File Reference

Implementation of an adapter of a DLL+PLL tracking loop block using GPU accelerated functions for GPS L1 C/A to a TrackingInterface.

```
#include "gps_l1_ca_dll_pll_tracking_gpu_cc.h"
#include "tracking_interface.h"
#include <string>
```

Classes

- class GpsL1CaDllPllTrackingGPU

  This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions.

13.483.1 Detailed Description

Implementation of an adapter of a DLL+PLL tracking loop block using GPU accelerated functions for GPS L1 C/A to a TrackingInterface.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_dll_pll_tracking_gpu.h.

13.484 gps_l1_ca_dll_pll_tracking_gpu.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_dll_pll_tracking_gpu.h
00003  * \brief Implementation of an adapter of a DLL+PLL tracking loop block using GPU accelerated
      functions
00004  * for GPS L1 C/A to a TrackingInterface
00005  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00006  *
00007  * Code DLL + carrier PLL according to the algorithms described in:
00008  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00009  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00010  * Approach, Birkhauser, 2007
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_H
00024 #define GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_H
00025
00026 #include "gps_l1_ca_dll_pll_tracking_gpu_cc.h"
00027 #include "tracking_interface.h"
00028 #include <string>
00029
00030 /** \addtogroup Tracking
00031  * \{ */
00032 /** \addtogroup Tracking_adapters
00033  * \{ */
00034
00035
00036 class ConfigurationInterface;
00037
00038 /*!
00039  * \brief This class implements a code DLL + carrier PLL tracking loop using GPU accelerated functions
00040  */
00041 class GpsL1CaDllPllTrackingGPU : public TrackingInterface
00042 {
00043 public:
00044     GpsL1CaDllPllTrackingGPU(
00045         const ConfigurationInterface* configuration,
00046         std::string role,
00047         unsigned int in_streams,
00048         unsigned int out_streams);
00049
00050     virtual ~GpsL1CaDllPllTrackingGPU();
00051
00052     inline std::string role() override
00053     {
00054         return role_;
00055     }
00056
00057     //! Returns "GPS_L1_CA_DLL_PLL_Tracking_GPU"
00058     inline std::string implementation() override
00059     {
00060         return "GPS_L1_CA_DLL_PLL_Tracking_GPU";
00061     }
00062
00063     inline size_t item_size() override
00064     {
00065         return item_size_;
00066     }
00067
00068     void connect(gr::top_block_sptr top_block) override;
00069     void disconnect(gr::top_block_sptr top_block) override;
00070     gr::basic_block_sptr get_left_block() override;
00071     gr::basic_block_sptr get_right_block() override;
00072
00073     /*!
00074      * \brief Set tracking channel unique ID
00075      */
00076     void set_channel(unsigned int channel) override;
00077
00078     /*!
00079      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00080      * to efficiently exchange synchronization data between acquisition and tracking blocks
00081      */
00082     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00083
00084     void start_tracking() override;
00085
00086     /*!
00087      * \brief Stop running tracking
00088      */
00089     void stop_tracking() override;
00090
00091 private:
00092     gps_l1_ca_dll_pll_tracking_gpu_cc_sptr tracking_;
00093     size_t item_size_;
00094     unsigned int channel_;
00095     std::string role_;
00096     unsigned int in_streams_;
00097     unsigned int out_streams_;
00098 };
00099
00100
00101 /** \} */
```

```
00102 /** \} */
00103 #endif  // GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_H
```

13.485 gps_l1_ca_gaussian_tracking.h File Reference

Interface of an adapter of a DLL + Kalman carrier tracking loop block for GPS L1 C/A signals.

```
#include "gps_l1_ca_gaussian_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

Classes

- class GpsL1CaGaussianTracking

  This class implements a code DLL + carrier PLL tracking loop.

13.485.1 Detailed Description

Interface of an adapter of a DLL + Kalman carrier tracking loop block for GPS L1 C/A signals.

Author

Javier Arribas, 2018. jarribas(at)cttc.es

Jordi Vila-Valls 2018. jvila(at)cttc.es

Carles Fernandez-Prades 2018. cfernandez(at)cttc.es

Reference: J. Vila-Valls, P. Closas, M. Navarro and C. Fernandez-Prades, "Are PLLs Dead? A Tutorial on Kalman Filter-based Techniques for Digital Carrier Synchronization", IEEE Aerospace and Electronic Systems Magazine, Vol. 32, No. 7, pp. 28–45, July 2017. DOI: 10.1109/MAES.2017.150260

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_gaussian_tracking.h.

13.486 gps_l1_ca_gaussian_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_gaussian_tracking.h
00003  * \brief Interface of an adapter of a DLL + Kalman carrier
00004  * tracking loop block for GPS L1 C/A signals
00005  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00006  * \author Jordi Vila-Valls 2018. jvila(at)cttc.es
00007  * \author Carles Fernandez-Prades 2018. cfernandez(at)cttc.es
00008  *
00009  * Reference:
00010  * J. Vila-Valls, P. Closas, M. Navarro and C. Fernandez-Prades,
00011  * "Are PLLs Dead? A Tutorial on Kalman Filter-based Techniques for Digital
00012  * Carrier Synchronization", IEEE Aerospace and Electronic Systems Magazine,
00013  * Vol. 32, No. 7, pp. 28-45, July 2017. DOI: 10.1109/MAES.2017.150260
00014  *
00015  * -----------------------------------------------------------------------------
00016  *
00017  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00018  * This file is part of GNSS-SDR.
00019  *
00020  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00021  * SPDX-License-Identifier: GPL-3.0-or-later
00022  *
00023  * -----------------------------------------------------------------------------
00024  */
00025
00026 #ifndef GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_H
00027 #define GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_H
00028
00029 #include "gps_l1_ca_gaussian_tracking_cc.h"
00030 #include "tracking_interface.h"
00031 #include <string>
00032
00033 /** \addtogroup Tracking
```

```
00034  * \{ */
00035 /** \addtogroup Tracking_adapters
00036  * \{ */
00037
00038
00039 class ConfigurationInterface;
00040
00041 /*!
00042  * \brief This class implements a code DLL + carrier PLL tracking loop
00043  */
00044 class GpsL1CaGaussianTracking : public TrackingInterface
00045 {
00046 public:
00047     GpsL1CaGaussianTracking(
00048         const ConfigurationInterface* configuration,
00049         const std::string& role,
00050         unsigned int in_streams,
00051         unsigned int out_streams);
00052
00053     ~GpsL1CaGaussianTracking() = default;
00054
00055     inline std::string role() override
00056     {
00057         return role_;
00058     }
00059
00060     //! Returns "GPS_L1_CA_Gaussian_Tracking"
00061     inline std::string implementation() override
00062     {
00063         return "GPS_L1_CA_Gaussian_Tracking";
00064     }
00065
00066     inline size_t item_size() override
00067     {
00068         return item_size_;
00069     }
00070
00071     void connect(gr::top_block_sptr top_block) override;
00072     void disconnect(gr::top_block_sptr top_block) override;
00073     gr::basic_block_sptr get_left_block() override;
00074     gr::basic_block_sptr get_right_block() override;
00075
00076     /*!
00077      * \brief Set tracking channel unique ID
00078      */
00079     void set_channel(unsigned int channel) override;
00080
00081     /*!
00082      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00083      * to efficiently exchange synchronization data between acquisition and tracking blocks
00084      */
00085     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00086
00087     void start_tracking() override;
00088
00089     /*!
00090      * \brief Stop running tracking
00091      */
00092     void stop_tracking() override;
00093
00094 private:
00095     gps_l1_ca_gaussian_tracking_cc_sptr tracking_sptr_;
00096     std::string role_;
00097     size_t item_size_;
00098     unsigned int channel_;
00099     unsigned int in_streams_;
00100     unsigned int out_streams_;
00101 };
00102
00103
00104 /** \} */
00105 /** \} */
00106 #endif  // GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_H
```

13.487 gps_l1_ca_kf_tracking.h File Reference

Interface of an adapter of a code + carrier Kalman Filter tracking loop with VTL capabilities block for GPS L1 C/A to a TrackingInterface.

```
#include "kf_tracking.h"
#include "tracking_interface.h"
#include <string>
```

Classes

- class GpsL1CaKfTracking

  This class implements a code + carrier Kalman Filter tracking loop with VTL capabilities.

13.487.1 Detailed Description

Interface of an adapter of a code + carrier Kalman Filter tracking loop with VTL capabilities block for GPS L1 C/A to a TrackingInterface.

Author

Javier Arribas, 2020. jarribas(at)cttc.es

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors)
GNSS-SDR is a software defined Global Navigation Satellite Systems receiver
This file is part of GNSS-SDR.
SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_kf_tracking.h.

13.488 gps_l1_ca_kf_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_kf_tracking.h
00003  * \brief  Interface of an adapter of a code + carrier Kalman Filter tracking
00004  * loop with VTL capabilities block
00005  * for GPS L1 C/A to a TrackingInterface
00006  * \author  Javier Arribas, 2020. jarribas(at)cttc.es
00007  *
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  *
00013  * GNSS-SDR is a software defined Global Navigation
00014  *          Satellite Systems receiver
00015  *
00016  * This file is part of GNSS-SDR.
00017  *
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_GPS_L1_CA_KF_TRACKING_H
00024 #define GNSS_SDR_GPS_L1_CA_KF_TRACKING_H
00025
00026 #include "kf_tracking.h"
00027 #include "tracking_interface.h"
00028 #include <string>
00029
00030 class ConfigurationInterface;
00031
00032 /*!
00033  * \brief This class implements a code + carrier Kalman Filter tracking loop
00034  * with VTL capabilities
00035  */
00036 class GpsL1CaKfTracking : public TrackingInterface
00037 {
00038 public:
00039     GpsL1CaKfTracking(
00040         const ConfigurationInterface* configuration,
00041         const std::string& role,
00042         unsigned int in_streams,
00043         unsigned int out_streams);
00044
00045     ~GpsL1CaKfTracking() = default;
00046
00047     inline std::string role() override
00048     {
00049         return role_;
00050     }
00051
00052     //! Returns "GPS_L1_CA_KF_Tracking"
00053     inline std::string implementation() override
00054     {
00055         return "GPS_L1_CA_KF_Tracking";
```

```
00056     }
00057
00058     inline size_t item_size() override
00059     {
00060         return item_size_;
00061     }
00062
00063     void connect(gr::top_block_sptr top_block) override;
00064     void disconnect(gr::top_block_sptr top_block) override;
00065     gr::basic_block_sptr get_left_block() override;
00066     gr::basic_block_sptr get_right_block() override;
00067
00068     /*!
00069      * \brief Set tracking channel unique ID
00070      */
00071     void set_channel(unsigned int channel) override;
00072
00073     /*!
00074      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00075      * to efficiently exchange synchronization data between acquisition
00076      * and tracking blocks
00077      */
00078     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
00079
00080     void start_tracking() override;
00081
00082     /*!
00083      * \brief Stop running tracking
00084      */
00085     void stop_tracking() override;
00086
00087 private:
00088     kf_tracking_sptr tracking_sptr_;
00089     std::string role_;
00090     size_t item_size_;
00091     unsigned int channel_;
00092     unsigned int in_streams_;
00093     unsigned int out_streams_;
00094 };
00095
00096 #endif  // GNSS_SDR_GPS_L1_CA_KF_TRACKING_H
```

13.489 gps_l1_ca_tcp_connector_tracking.h File Reference

Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for GPS L1 C/A to a TrackingInterface.

```
#include "gps_l1_ca_tcp_connector_tracking_cc.h"
#include "tracking_interface.h"
#include <string>
```

Classes

- class GpsL1CaTcpConnectorTracking

  This class implements a code DLL + carrier PLL tracking loop.

13.489.1 Detailed Description

Interface of an adapter of a TCP connector block based on code DLL + carrier PLL for GPS L1 C/A to a TrackingInterface.

Author

David Pubill, 2012. dpubill(at)cttc.es Javier Arribas, 2011. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_tcp_connector_tracking.h.

13.490 gps_l1_ca_tcp_connector_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_tcp_connector_tracking.h
00003  * \brief  Interface of an adapter of a TCP connector block based on code DLL + carrier PLL
00004  * for GPS L1 C/A to a TrackingInterface
00005  * \author David Pubill, 2012. dpubill(at)cttc.es
00006  *         Javier Arribas, 2011. jarribas(at)cttc.es
00007  *
00008  * Code DLL + carrier PLL according to the algorithms described in:
00009  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00010  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency
00011  * Approach, Birkhauser, 2007
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_H
00025 #define GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_H
00026
00027 #include "gps_l1_ca_tcp_connector_tracking_cc.h"
00028 #include "tracking_interface.h"
00029 #include <string>
00030
00031 /** \addtogroup Tracking
00032  * \{ */
00033 /** \addtogroup Tracking_adapters
00034  * \{ */
00035
00036
00037 class ConfigurationInterface;
00038
00039 /*!
00040  * \brief This class implements a code DLL + carrier PLL tracking loop
00041  */
00042 class GpsL1CaTcpConnectorTracking : public TrackingInterface
00043 {
00044 public:
00045     GpsL1CaTcpConnectorTracking(
00046         const ConfigurationInterface* configuration,
00047         const std::string& role,
00048         unsigned int in_streams,
00049         unsigned int out_streams);
00050
00051     ~GpsL1CaTcpConnectorTracking() = default;
00052
00053     inline std::string role() override
00054     {
00055         return role_;
00056     }
00057
00058     //! Returns "GPS_L1_CA_TCP_CONNECTOR_Tracking"
00059     inline std::string implementation() override
00060     {
00061         return "GPS_L1_CA_TCP_CONNECTOR_Tracking";
00062     }
00063
00064     inline size_t item_size() override
00065     {
00066         return item_size_;
00067     }
00068
00069     void connect(gr::top_block_sptr top_block) override;
00070     void disconnect(gr::top_block_sptr top_block) override;
00071     gr::basic_block_sptr get_left_block() override;
00072     gr::basic_block_sptr get_right_block() override;
00073
00074     /*!
00075      * \brief Set tracking channel unique ID
00076      */
00077     void set_channel(unsigned int channel) override;
00078
00079     /*!
00080      * \brief Set acquisition/tracking common Gnss_Synchro object pointer
00081      * to efficiently exchange synchronization data between acquisition and tracking blocks
00082      */
00083     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro) override;
```

```
00084
00085     void start_tracking() override;
00086
00087     /*!
00088      * \brief Stop running tracking
00089      */
00090     void stop_tracking() override;
00091
00092 private:
00093     gps_l1_ca_tcp_connector_tracking_cc_sptr tracking_sptr_;
00094     std::string role_;
00095     size_t item_size_;
00096     unsigned int channel_;
00097     unsigned int in_streams_;
00098     unsigned int out_streams_;
00099 };
00100
00101
00102 /** \} */
00103 /** \} */
00104 #endif  // GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_H
```

13.491 dll_pll_veml_tracking.h File Reference

Implementation of a code DLL + carrier PLL tracking block.

```
#include "bit_synchronizer.h"
#include "cpu_multicorrelator_real_codes.h"
#include "dll_pll_conf.h"
#include "exponential_smoother.h"
#include "gnss_block_interface.h"
#include "gnss_time.h"
#include "tow_to_trk.h"
#include "tracking_FLL_PLL_filter.h"
#include "tracking_loop_filter.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <string>
#include <typeinfo>
#include <utility>
#include <vector>
```

Classes

- class dll_pll_veml_tracking

  This class implements a code DLL + carrier PLL tracking block.

Typedefs

- using dll_pll_veml_tracking_sptr = gnss_shared_ptr<dll_pll_veml_tracking>

Functions

- dll_pll_veml_tracking_sptr **dll_pll_veml_make_tracking** (const Dll_Pll_Conf &conf_)

13.491.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block.

Author

Javier Arribas, 2018-2025. jarribas(at)cttc.es

Carles Fernandez-Prades, 2018-2025 carles.fernandez(at)cttc.es

Antonio Ramos, 2018 antonio.ramosdet(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dll_pll_veml_tracking.h.

13.492 dll_pll_veml_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_veml_tracking.h
00003  * \brief Implementation of a code DLL + carrier PLL tracking block.
00004  * \author Javier Arribas, 2018-2025. jarribas(at)cttc.es
00005  * \author Carles Fernandez-Prades, 2018-2025 carles.fernandez(at)cttc.es
00006  * \author Antonio Ramos, 2018 antonio.ramosdet(at)gmail.com
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DLL_PLL_VEML_TRACKING_H
00020 #define GNSS_SDR_DLL_PLL_VEML_TRACKING_H
00021
00022 #include "bit_synchronizer.h"
00023 #include "cpu_multicorrelator_real_codes.h"
00024 #include "dll_pll_conf.h"
00025 #include "exponential_smoother.h"
00026 #include "gnss_block_interface.h"
00027 #include "gnss_time.h"  // for timetags produced by File_Timestamp_Signal_Source
00028 #include "tow_to_trk.h"
00029 #include "tracking_FLL_PLL_filter.h"  // for PLL/FLL filter
00030 #include "tracking_loop_filter.h"     // for DLL filter
00031 #include <boost/circular_buffer.hpp>
00032 #include <gnuradio/block.h>                  // for block
00033 #include <gnuradio/gr_complex.h>             // for gr_complex
00034 #include <gnuradio/types.h>                  // for gr_vector_int, gr_vector...
00035 #include <pmt/pmt.h>                         // for pmt_t
00036 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00037 #include <cstddef>                           // for size_t
00038 #include <cstdint>                           // for int32_t
00039 #include <fstream>                           // for ofstream
00040 #include <string>                            // for string
00041 #include <typeinfo>                          // for typeid
00042 #include <utility>                           // for pair
00043 #include <vector>                            // for vector
00044
00045 /** \addtogroup Tracking
00046  * \{ */
00047 /** \addtogroup Tracking_gnuradio_blocks tracking_gr_blocks
00048  * GNU Radio blocks for GNSS signal tracking.
00049  * \{ */
00050
00051
00052 class Gnss_Synchro;
00053 class dll_pll_veml_tracking;
00054
00055 using dll_pll_veml_tracking_sptr = gnss_shared_ptr<dll_pll_veml_tracking>;
00056
00057 dll_pll_veml_tracking_sptr dll_pll_veml_make_tracking(const Dll_Pll_Conf &conf_);
00058
00059 /*!
00060  * \brief This class implements a code DLL + carrier PLL tracking block.
00061  */
00062 class dll_pll_veml_tracking : public gr::block
00063 {
00064 public:
00065     ~dll_pll_veml_tracking() override;
00066
00067     void set_channel(uint32_t channel);
00068     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
```

```
00069     void start_tracking();
00070     void stop_tracking();
00071
00072     int general_work(int noutput_items, gr_vector_int &ninput_items,
00073         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items) override;
00074
00075     void forecast(int noutput_items, gr_vector_int &ninput_items_required) override;
00076
00077     //! Doppler offset multiplier (in units of f_error_doppler_step) for the given
00078     //! frequency-error-reduction bin index: bin 0 -> 0, then alternating outward
00079     //! +1, -1, +2, -2, +3, -3, ...
00080     static double f_error_bin_multiplier(uint32_t bin_index);
00081
00082 private:
00083     friend class DllPllTrackingFrequencyErrorTest;
00084     friend dll_pll_veml_tracking_sptr dll_pll_veml_make_tracking(const Dll_Pll_Conf &conf_);
00085     explicit dll_pll_veml_tracking(const Dll_Pll_Conf &conf_);
00086
00087     void msg_handler_telemetry_to_trk(const pmt::pmt_t &msg);
00088     void do_correlation_step(const gr_complex *input_samples);
00089     void run_dll_pll();
00090     // State 5: frequency-error-reduction Doppler-bin scan, run right after pull-in (state 1)
00091     // and before wide tracking (state 2) -- see general_work()'s case 5. Fully passive: drives
00092     // only d_carrier_doppler_hz from the swept test bin; run_dll_pll() never runs during the
00093     // scan, so code phase and the carrier loop filter's state stay exactly as pull-in left
00094     // them for the scan's whole duration.
00095     void run_f_error_scan_step();
00096     void begin_wide_tracking(uint64_t sample_count);
00097     uint64_t tracking_elapsed_seconds(uint64_t sample_count) const;
00098     void check_carrier_phase_coherent_initialization();
00099     void update_tracking_vars(bool estimate_rate = true);
00100     void clear_tracking_vars();
00101     void save_correlation_results();
00102     void log_data();
00103     void map_correlator_to_iq(const gr_complex &c, float &out_i, float &out_q) const;
00104     void assign_correlators_to_synchro(Gnss_Synchro &synchro) const;
00105     void configure_bit_synchronizer();
00106     bool cn0_and_tracking_lock_status(double coh_integration_time_s);
00107     bool acquire_secondary();
00108     int64_t uint64diff(uint64_t first, uint64_t second);
00109     int32_t save_matfile() const;
00110
00111     Cpu_Multicorrelator_Real_Codes d_multicorrelator_cpu;
00112
00113     Dll_Pll_Conf d_trk_parameters;
00114
00115     Exponential_Smoother d_cn0_smoother;
00116     Exponential_Smoother d_carrier_lock_test_smoother;
00117
00118     Tracking_loop_filter d_code_loop_filter;
00119     Tracking_FLL_PLL_filter d_carrier_loop_filter;
00120
00121     HistogramBitSynchronizer d_bit_sync;
00122
00123     Gnss_Synchro *d_acquisition_gnss_synchro;
00124
00125     volk_gnsssdr::vector<float> d_tracking_code;
00126     volk_gnsssdr::vector<float> d_data_code;
00127     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00128     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00129     volk_gnsssdr::vector<gr_complex> d_Prompt_Data;
00130     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00131
00132     boost::circular_buffer<float> d_dll_filt_history;
00133     boost::circular_buffer<std::pair<double, double» d_code_ph_history;
00134     boost::circular_buffer<std::pair<double, double» d_carr_ph_history;
00135     boost::circular_buffer<gr_complex> d_Prompt_circular_buffer;
00136
00137     const size_t d_int_type_hash_code = typeid(int).hash_code();
00138     const size_t d_tow_to_trk_type_hash_code = typeid(std::shared_ptr<TOW_to_trk>).hash_code();
00139
00140     double d_signal_carrier_freq;
00141     double d_code_period;
00142     double d_code_chip_rate;
00143     double d_acq_code_phase_samples;
00144     double d_acq_carrier_doppler_hz;
00145     double d_current_correlation_time_s;
00146     double d_carr_phase_error_hz;
00147     double d_carr_freq_error_hz;
00148     double d_carr_error_filt_hz;
00149     double d_code_error_chips;
00150     double d_code_error_filt_chips;
00151     double d_code_freq_chips;
00152     double d_cfo_frequency_hz;
00153     double d_cfo_phase_step_rad;
00154     double d_carrier_doppler_hz;
00155     double d_acc_carrier_phase_rad;
```

```
00156     double d_rem_code_phase_chips;
00157     double d_T_chip_seconds;
00158     double d_T_prn_seconds;
00159     double d_T_prn_samples;
00160     double d_K_blk_samples;
00161     double d_carrier_lock_test;
00162     double d_CN0_SNV_dB_Hz;
00163     double d_carrier_lock_threshold;
00164     double d_carrier_phase_step_rad;
00165     double d_carrier_phase_rate_step_rad;
00166     double d_code_phase_step_chips;
00167     double d_code_phase_rate_step_chips;
00168     double d_rem_code_phase_samples;
00169
00170     gr_complex *d_Very_Early;
00171     gr_complex *d_Early;
00172     gr_complex *d_Prompt;
00173     gr_complex *d_Late;
00174     gr_complex *d_Very_Late;
00175
00176     gr_complex d_VE_accu;
00177     gr_complex d_E_accu;
00178     gr_complex d_P_accu;
00179     gr_complex d_P_accu_old;
00180     gr_complex d_L_accu;
00181     gr_complex d_VL_accu;
00182     gr_complex d_P_data_accu;
00183
00184     std::string d_secondary_code_string;
00185     std::string d_data_secondary_code_string;
00186     std::string d_systemName;
00187     std::string d_signal_type;
00188     std::string d_signal_pretty_name;
00189     std::string d_dump_filename;
00190
00191     std::ofstream d_dump_file;
00192
00193     // uint64_t d_sample_counter;
00194     uint64_t d_acq_sample_stamp;
00195     uint64_t d_tracking_time_start_sample{0};
00196     GnssTime d_last_timetag{};
00197     std::shared_ptr<TOW_to_trk> d_last_tow_received;
00198     uint64_t d_last_timetag_samplecounter;
00199     bool d_timetag_waiting;
00200
00201     float *d_prompt_data_shift;
00202     float d_rem_carr_phase_rad;
00203
00204     uint64_t d_tow_from_telemetry_ms{};
00205     int64_t d_bit_sync_target_epoch{};
00206     int32_t d_wn_from_telemetry{};
00207
00208     int32_t d_symbols_per_bit;
00209     int32_t d_state;
00210     int32_t d_correlation_length_ms;
00211     int32_t d_n_correlator_taps;
00212     int32_t d_current_prn_length_samples;
00213     int32_t d_extend_correlation_symbols_count;
00214     int32_t d_extend_correlation_symbols;
00215     int32_t d_current_symbol;
00216     int32_t d_current_data_symbol;
00217     int32_t d_cn0_estimation_counter;
00218     int32_t d_carrier_lock_fail_counter;
00219     int32_t d_code_lock_fail_counter;
00220     int32_t d_code_samples_per_chip;  // All signals have 1 sample per chip code except Gal. E1 which
      has 2 (CBOC disabled) or 12 (CBOC enabled)
00221     int32_t d_code_length_chips;
00222
00223     uint32_t d_channel;
00224     uint32_t d_secondary_code_length;
00225     uint32_t d_data_secondary_code_length;
00226
00227     // State 5: frequency error reduction (passive Doppler bin scan run right after pull-in).
00228     // Bin count comes from d_trk_parameters.f_error_step_num (0 disables state 5 entirely).
00229     uint32_t d_f_error_num_bins;
00230     uint32_t d_f_error_bin_index;
00231     uint32_t d_f_error_accum_counter;
00232     double d_f_error_center_doppler_hz;
00233     std::vector<double> d_f_error_power;
00234     // Raw per-bin Prompt correlator samples (bin index -> f_error_accumulation samples),
00235     // kept purely so cn0_m2m4_estimator() can report a CN0 estimate for the winning bin
00236     // in the frequency-error-reduction diagnostics; not used for anything else.
00237     std::vector<std::vector<gr_complex» d_f_error_prompt_samples;
00238     // Doppler the frequency-error-reduction scan selected at the end of its run (see
00239     // run_f_error_scan_step()), kept separately from d_carrier_doppler_hz (which gets
00240     // overwritten by the FLL/PLL every epoch afterward) so a subsequent loss-of-lock can
00241     // report how far steady-state tracking has drifted from what the scan originally chose
```

```
00242     // -- e.g. to check whether the loop settled on a false/sidelobe equilibrium.
00243     double d_f_error_selected_doppler_hz = 0.0;
00244
00245     bool d_pull_in_transitory;
00246     bool d_corrected_doppler;
00247     bool d_interchange_iq;
00248     bool d_veml;
00249     bool d_cloop;
00250     bool d_secondary;
00251     bool d_dump;
00252     bool d_dump_mat;
00253     bool d_acc_carrier_phase_initialized;
00254     bool d_enable_extended_integration;
00255     bool d_Flag_PLL_180_deg_phase_locked;
00256     bool d_use_histogram_bit_sync;
00257     bool d_wait_for_bit_edge{false};
00258     bool d_b1c_prelock_output_pending{false};
00259     bool d_carrier_phase_discontinuity{true};  // pending report of a new carrier phase ambiguity
00260 };
00261
00262
00263 /** \} */
00264 /** \} */
00265 #endif  // GNSS_SDR_DLL_PLL_VEML_TRACKING_H
```

13.493 dll_pll_veml_tracking_fpga.h File Reference

Implementation of a code DLL + carrier PLL tracking block using an FPGA.

```
#include "bit_synchronizer.h"
#include "dll_pll_conf_fpga.h"
#include "exponential_smoother.h"
#include "gnss_block_interface.h"
#include "tracking_FLL_PLL_filter.h"
#include "tracking_loop_filter.h"
#include <boost/circular_buffer.hpp>
#include <gnuradio/block.h>
#include <gnuradio/gr_complex.h>
#include <gnuradio/types.h>
#include <pmt/pmt.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstddef>
#include <cstdint>
#include <fstream>
#include <memory>
#include <string>
#include <typeinfo>
#include <utility>
```

Classes

- class dll_pll_veml_tracking_fpga

  This class implements a code DLL + carrier PLL tracking block.

Typedefs

- using dll_pll_veml_tracking_fpga_sptr = gnss_shared_ptr<dll_pll_veml_tracking_fpga>

Functions

- dll_pll_veml_tracking_fpga_sptr **dll_pll_veml_make_tracking_fpga** (const Dll_Pll_Conf_Fpga &conf_)

13.493.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block using an FPGA.

Author

Marc Majoral, 2019. marc.majoral(at)cttc.es

Javier Arribas, 2019. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dll_pll_veml_tracking_fpga.h.

13.494 dll_pll_veml_tracking_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_veml_tracking_fpga.h
00003  * \brief Implementation of a code DLL + carrier PLL tracking block using an FPGA.
00004  * \author Marc Majoral, 2019. marc.majoral(at)cttc.es
00005  * \author Javier Arribas, 2019. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_DLL_PLL_VEML_TRACKING_FPGA_H
00019 #define GNSS_SDR_DLL_PLL_VEML_TRACKING_FPGA_H
00020
00021 #include "bit_synchronizer.h"
00022 #include "dll_pll_conf_fpga.h"
00023 #include "exponential_smoother.h"
00024 #include "gnss_block_interface.h"
00025 #include "tracking_FLL_PLL_filter.h"  // for PLL/FLL filter
00026 #include "tracking_loop_filter.h"     // for DLL filter
00027 #include <boost/circular_buffer.hpp>
00028 #include <gnuradio/block.h>                    // for block
00029 #include <gnuradio/gr_complex.h>               // for gr_complex
00030 #include <gnuradio/types.h>                    // for gr_vector_int, gr_vector...
00031 #include <pmt/pmt.h>                           // for pmt_t
00032 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>   // for volk_gnsssdr::vector
00033 #include <cstddef>                             // for size_t
00034 #include <cstdint>                             // for int32_t
00035 #include <fstream>                             // for string, ofstream
00036 #include <memory>                              // for std::shared_ptr
00037 #include <string>                              // for string
00038 #include <typeinfo>                            // for typeid
00039 #include <utility>                             // for pair
00040
00041 /** \addtogroup Tracking
00042  * \{ */
00043 /** \addtogroup Tracking_gnuradio_blocks
00044  * \{ */
00045
00046
00047 class Fpga_Multicorrelator_8sc;
00048 class Gnss_Synchro;
00049 class dll_pll_veml_tracking_fpga;
00050
00051 using dll_pll_veml_tracking_fpga_sptr = gnss_shared_ptr<dll_pll_veml_tracking_fpga>;
00052
00053 dll_pll_veml_tracking_fpga_sptr dll_pll_veml_make_tracking_fpga(const Dll_Pll_Conf_Fpga &conf_);
00054
00055
00056 /*!
00057  * \brief This class implements a code DLL + carrier PLL tracking block.
00058  */
00059 class dll_pll_veml_tracking_fpga : public gr::block
00060 {
00061 public:
00062     /*!
00063      * \brief Destructor
00064      */
00065     ~dll_pll_veml_tracking_fpga();
00066
00067     /*!
00068      * \brief Set the channel number and configure some multicorrelator parameters
00069      */
00070     void set_channel(uint32_t channel, const std::string &device_io_name);
```

```
00071
00072     /*!
00073      * \brief This function is used with two purposes:
00074      * 1 -> To set the gnss_synchro
00075      * 2 -> A set_gnss_synchro command with a valid PRN is received when the system is going to run
00076      * acquisition with that PRN. We can use this command to pre-initialize tracking parameters and
00077      * variables before the actual acquisition process takes place. In this way we minimize the
00078      * latency between acquisition and tracking once the acquisition has been made.
00079      */
00080     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
00081
00082     /*!
00083      * \brief This function starts the tracking process
00084      */
00085     void start_tracking();
00086
00087     /*!
00088      * \brief This function sets a flag that makes general_work to stop in order to finish the
      tracking process.
00089      */
00090     void stop_tracking();
00091
00092     /*!
00093      * \brief General Work
00094      */
00095     int general_work(int noutput_items, gr_vector_int &ninput_items,
00096         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00097
00098     /*!
00099      * \brief This function disables the HW multicorrelator in the FPGA in order to stop the tracking
      process
00100      */
00101     void reset();
00102
00103 private:
00104     friend dll_pll_veml_tracking_fpga_sptr dll_pll_veml_make_tracking_fpga(const Dll_Pll_Conf_Fpga
      &conf_);
00105     explicit dll_pll_veml_tracking_fpga(const Dll_Pll_Conf_Fpga &conf_);
00106
00107     void msg_handler_telemetry_to_trk(const pmt::pmt_t &msg);
00108     bool cn0_and_tracking_lock_status(double coh_integration_time_s);
00109     bool acquire_secondary();
00110     void do_correlation_step();
00111     void run_dll_pll();
00112     void check_carrier_phase_coherent_initialization();
00113     void update_tracking_vars();
00114     void clear_tracking_vars();
00115     void save_correlation_results();
00116     void log_data();
00117     void configure_bit_synchronizer();
00118     int32_t save_matfile() const;
00119
00120     Dll_Pll_Conf_Fpga d_trk_parameters;
00121     HistogramBitSynchronizer d_bit_sync;
00122     Exponential_Smoother d_cn0_smoother;
00123     Exponential_Smoother d_carrier_lock_test_smoother;
00124
00125     Gnss_Synchro *d_acquisition_gnss_synchro;
00126
00127     Tracking_loop_filter d_code_loop_filter;
00128
00129     Tracking_FLL_PLL_filter d_carrier_loop_filter;
00130
00131     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00132     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00133     volk_gnsssdr::vector<gr_complex> d_Prompt_Data;
00134     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00135
00136     boost::circular_buffer<float> d_dll_filt_history;
00137     boost::circular_buffer<std::pair<double, double» d_code_ph_history;
00138     boost::circular_buffer<std::pair<double, double» d_carr_ph_history;
00139     boost::circular_buffer<gr_complex> d_Prompt_circular_buffer;
00140
00141     std::string d_systemName;
00142     std::string d_signal_type;
00143     std::string d_secondary_code_string;
00144     std::string d_data_secondary_code_string;
00145     std::string d_signal_pretty_name;
00146     std::string d_dump_filename;
00147
00148     std::ofstream d_dump_file;
00149
00150     std::shared_ptr<Fpga_Multicorrelator_8sc> d_multicorrelator_fpga;
00151
00152     boost::condition_variable d_m_condition;
00153
00154     boost::mutex d_mutex;
```

```
00155
00156     const size_t int_type_hash_code = typeid(int).hash_code();
00157
00158     double d_signal_carrier_freq;
00159     double d_code_period;
00160     double d_code_chip_rate;
00161     double d_code_phase_step_chips;
00162     double d_code_phase_rate_step_chips;
00163     double d_carrier_phase_step_rad;
00164     double d_carrier_phase_rate_step_rad;
00165     double d_acq_code_phase_samples;
00166     double d_acq_carrier_doppler_hz;
00167     double d_rem_code_phase_samples;
00168     double d_rem_code_phase_samples_prev;
00169     double d_current_correlation_time_s;
00170     double d_carr_phase_error_hz;
00171     double d_carr_freq_error_hz;
00172     double d_carr_error_filt_hz;
00173     double d_code_error_chips;
00174     double d_code_error_filt_chips;
00175     double d_code_freq_chips;
00176     double d_carrier_doppler_hz;
00177     double d_acc_carrier_phase_rad;
00178     double d_rem_code_phase_chips;
00179     double d_T_chip_seconds;
00180     double d_T_prn_seconds;
00181     double d_T_prn_samples;
00182     double d_K_blk_samples;
00183     double d_carrier_lock_test;
00184     double d_CN0_SNV_dB_Hz;
00185     double d_carrier_lock_threshold;
00186
00187     gr_complex *d_Very_Early;
00188     gr_complex *d_Early;
00189     gr_complex *d_Prompt;
00190     gr_complex *d_Late;
00191     gr_complex *d_Very_Late;
00192
00193     gr_complex d_VE_accu;
00194     gr_complex d_E_accu;
00195     gr_complex d_P_accu;
00196     gr_complex d_P_accu_old;
00197     gr_complex d_L_accu;
00198     gr_complex d_VL_accu;
00199     gr_complex d_P_data_accu;
00200
00201     uint64_t d_sample_counter;
00202     uint64_t d_acq_sample_stamp;
00203     uint64_t d_sample_counter_next;
00204     int64_t d_bit_sync_target_epoch{};
00205
00206     float *d_prompt_data_shift;
00207     float d_rem_carr_phase_rad;
00208
00209     int32_t d_symbols_per_bit;
00210     int32_t d_state;
00211     int32_t d_extend_correlation_symbols_count;
00212     int32_t d_current_symbol;
00213     int32_t d_current_data_symbol;
00214     int32_t d_current_integration_length_samples;
00215     int32_t d_cn0_estimation_counter;
00216     int32_t d_carrier_lock_fail_counter;
00217     int32_t d_code_lock_fail_counter;
00218     int32_t d_correlation_length_ms;
00219     int32_t d_n_correlator_taps;
00220     int32_t d_next_integration_length_samples;
00221     int32_t d_extend_fpga_integration_periods;
00222
00223     uint32_t d_channel;
00224     uint32_t d_secondary_code_length;
00225     uint32_t d_data_secondary_code_length;
00226     uint32_t d_code_length_chips;
00227     uint32_t d_code_samples_per_chip;  // All signals have 1 sample per chip code except Gal. E1 which
      has 2 (CBOC disabled) or 12 (CBOC enabled)
00228     uint32_t d_fpga_integration_period;
00229     uint32_t d_current_fpga_integration_period;
00230
00231     bool d_veml;
00232     bool d_cloop;
00233     bool d_secondary;
00234     bool d_enable_extended_integration;
00235     bool d_dump;
00236     bool d_dump_mat;
00237     bool d_pull_in_transitory;
00238     bool d_corrected_doppler;
00239     bool d_interchange_iq;
00240     bool d_acc_carrier_phase_initialized;
```

```
00241     bool d_carrier_phase_discontinuity{true};  // pending report of a new carrier phase ambiguity
00242     bool d_worker_is_done;
00243     bool d_extended_correlation_in_fpga;
00244     bool d_current_extended_correlation_in_fpga;
00245     bool d_stop_tracking;
00246     bool d_sc_demodulate_enabled;
00247     bool d_Flag_PLL_180_deg_phase_locked;
00248     bool d_use_histogram_bit_sync;
00249     bool d_wait_for_bit_edge{false};
00250 };
00251
00252
00253 /** \} */
00254 /** \} */
00255 #endif  // GNSS_SDR_DLL_PLL_VEML_TRACKING_FPGA_H
```

13.495 galileo_e1_tcp_connector_tracking_cc.h File Reference

Interface of a TCP connector block based on code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.

```
#include "cpu_multicorrelator.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tcp_communication.h"
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

Classes

- class Galileo_E1_Tcp_Connector_Tracking_cc

  This class implements a code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.

Typedefs

- using galileo_e1_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Galileo_E1_Tcp_Connector_Tracking_cc>

Functions

- galileo_e1_tcp_connector_tracking_cc_sptr **galileo_e1_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float pll_bw_hz, float dll_bw_hz, float early_late_space_chips, float very_early_late_space_chips, size_t port_ch0)

13.495.1 Detailed Description

Interface of a TCP connector block based on code DLL + carrier PLL VEML (Very Early Minus Late) tracking block for Galileo E1 signals.

Author

David Pubill, 2012. dpubill(at)cttc.es Luis Esteve, 2012. luis(at)epsilon-formacion.com Javier Arribas, 2011. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_e1_tcp_connector_tracking_cc.h.

13.496 galileo_e1_tcp_connector_tracking_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e1_tcp_connector_tracking_cc.h
00003  * \brief Interface of a TCP connector block based on code DLL + carrier PLL VEML (Very Early
00004  *  Minus Late) tracking block for Galileo E1 signals
00005  * \author David Pubill, 2012. dpubill(at)cttc.es
00006  *         Luis Esteve, 2012. luis(at)epsilon-formacion.com
00007  *         Javier Arribas, 2011. jarribas(at)cttc.es
00008  *
00009  * Code DLL + carrier PLL according to the algorithms described in:
00010  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00011  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach,
00012  * Birkhauser, 2007
00013  *
00014  * -----------------------------------------------------------------------------
00015  *
00016  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00017  * This file is part of GNSS-SDR.
00018  *
00019  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00020  * SPDX-License-Identifier: GPL-3.0-or-later
00021  *
00022  * -----------------------------------------------------------------------------
00023  */
00024
00025 #ifndef GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_CC_H
00026 #define GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_CC_H
00027
00028 #include "cpu_multicorrelator.h"
00029 #include "gnss_block_interface.h"
00030 #include "gnss_synchro.h"
00031 #include "tcp_communication.h"
00032 #include <gnuradio/block.h>
00033 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00034 #include <fstream>
00035 #include <map>
00036 #include <string>
00037
00038 /** \addtogroup Tracking
00039  * \{ */
00040 /** \addtogroup Tracking_gnuradio_blocks
00041  * \{ */
00042
00043
00044 class Galileo_E1_Tcp_Connector_Tracking_cc;
00045
00046 using galileo_e1_tcp_connector_tracking_cc_sptr =
      gnss_shared_ptr<Galileo_E1_Tcp_Connector_Tracking_cc>;
00047
00048 galileo_e1_tcp_connector_tracking_cc_sptr
00049 galileo_e1_tcp_connector_make_tracking_cc(
00050     int64_t fs_in, uint32_t vector_length,
00051     bool dump,
00052     const std::string &dump_filename,
00053     float pll_bw_hz,
00054     float dll_bw_hz,
00055     float early_late_space_chips,
00056     float very_early_late_space_chips,
00057     size_t port_ch0);
00058
00059 /*!
00060  * \brief This class implements a code DLL + carrier PLL VEML (Very Early
00061  *  Minus Late) tracking block for Galileo E1 signals
00062  */
00063 class Galileo_E1_Tcp_Connector_Tracking_cc : public gr::block
00064 {
00065 public:
00066     ~Galileo_E1_Tcp_Connector_Tracking_cc();
00067
00068     void set_channel(uint32_t channel);
00069     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
00070     void start_tracking();
00071
00072     int general_work(int noutput_items, gr_vector_int &ninput_items,
00073         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00074
00075     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00076
00077 private:
00078     friend galileo_e1_tcp_connector_tracking_cc_sptr
00079     galileo_e1_tcp_connector_make_tracking_cc(
00080         int64_t fs_in, uint32_t vector_length,
00081         bool dump,
00082         const std::string &dump_filename,
```

```
00083         float pll_bw_hz,
00084         float dll_bw_hz,
00085         float early_late_space_chips,
00086         float very_early_late_space_chips,
00087         size_t port_ch0);
00088
00089     Galileo_E1_Tcp_Connector_Tracking_cc(
00090         int64_t fs_in, uint32_t vector_length,
00091         bool dump,
00092         const std::string &dump_filename,
00093         float pll_bw_hz,
00094         float dll_bw_hz,
00095         float early_late_space_chips,
00096         float very_early_late_space_chips,
00097         size_t port_ch0);
00098
00099     void update_local_code();
00100
00101     void update_local_carrier();
00102
00103     // tracking configuration vars
00104     uint32_t d_vector_length;
00105     bool d_dump;
00106
00107     Gnss_Synchro *d_acquisition_gnss_synchro;
00108     uint32_t d_channel;
00109
00110     int64_t d_fs_in;
00111
00112     int32_t d_correlation_length_samples;
00113     int32_t d_n_correlator_taps;
00114     float d_early_late_spc_chips;
00115     float d_very_early_late_spc_chips;
00116
00117     volk_gnsssdr::vector<gr_complex> d_ca_code;
00118
00119     gr_complex *d_Very_Early;
00120     gr_complex *d_Early;
00121     gr_complex *d_Prompt;
00122     gr_complex *d_Late;
00123     gr_complex *d_Very_Late;
00124
00125     // remaining code phase and carrier phase between tracking loops
00126     double d_rem_code_phase_samples;
00127     float d_next_rem_code_phase_samples;
00128     float d_rem_carr_phase_rad;
00129
00130     // acquisition
00131     float d_acq_code_phase_samples;
00132     float d_acq_carrier_doppler_hz;
00133
00134     // correlator
00135     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00136     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00137     Cpu_Multicorrelator multicorrelator_cpu;
00138
00139     // tracking vars
00140     double d_code_freq_chips;
00141     float d_carrier_doppler_hz;
00142     float d_acc_carrier_phase_rad;
00143     float d_acc_code_phase_secs;
00144     float d_code_phase_samples;
00145     size_t d_port_ch0;
00146     size_t d_port;
00147     int32_t d_listen_connection;
00148     float d_control_id;
00149     Tcp_Communication d_tcp_com;
00150
00151     // PRN period in samples
00152     int32_t d_current_prn_length_samples;
00153     int32_t d_next_prn_length_samples;
00154
00155     // processing samples counters
00156     uint64_t d_sample_counter;
00157     uint64_t d_acq_sample_stamp;
00158
00159     // CN0 estimation and lock detector
00160     int32_t d_cn0_estimation_counter;
00161     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00162     float d_carrier_lock_test;
00163     float d_CN0_SNV_dB_Hz;
00164     float d_carrier_lock_threshold;
00165     int32_t d_carrier_lock_fail_counter;
00166
00167     // control vars
00168     bool d_enable_tracking;
00169     bool d_pull_in;
```

```
00170
00171     // file dump
00172     std::string d_dump_filename;
00173     std::ofstream d_dump_file;
00174
00175     std::map<std::string, std::string> systemName;
00176     std::string sys;
00177 };
00178
00179
00180 /** \} */
00181 /** \} */
00182 #endif  // GNSS_SDR_GALILEO_E1_TCP_CONNECTOR_TRACKING_CC_H
```

13.497 gps_l1_ca_dll_pll_tracking_gpu_cc.h File Reference

Implementation of a code DLL + carrier PLL tracking block, GPU ACCELERATED.

```
#include "cuda_multicorrelator.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tracking_2nd_DLL_filter.h"
#include "tracking_FLL_PLL_filter.h"
#include <gnuradio/block.h>
#include <fstream>
#include <map>
#include <string>
#include <vector>
```

Classes

- class Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc

  This class implements a DLL + PLL tracking loop block.

Typedefs

- using gps_l1_ca_dll_pll_tracking_gpu_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc>

Functions

- gps_l1_ca_dll_pll_tracking_gpu_cc_sptr **gps_l1_ca_dll_pll_make_tracking_gpu_cc** (int64_t fs_in, uint32_t vector_length, bool dump, std::string dump_filename, float pll_bw_hz, float dll_bw_hz, float early_↩ late_space_chips)

13.497.1 Detailed Description

Implementation of a code DLL + carrier PLL tracking block, GPU ACCELERATED.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_dll_pll_tracking_gpu_cc.h.

13.498 gps_l1_ca_dll_pll_tracking_gpu_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_dll_pll_tracking_gpu_cc.h
```

```
00003  * \brief Implementation of a code DLL + carrier PLL tracking block, GPU ACCELERATED
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  * Code DLL + carrier PLL according to the algorithms described in:
00007  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00008  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach,
00009  * Birkhauser, 2007
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_CC_H
00023 #define GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_CC_H
00024
00025 #include "cuda_multicorrelator.h"
00026 #include "gnss_block_interface.h"
00027 #include "gnss_synchro.h"
00028 #include "tracking_2nd_DLL_filter.h"
00029 #include "tracking_FLL_PLL_filter.h"
00030 #include <gnuradio/block.h>
00031 #include <fstream>
00032 #include <map>
00033 #include <string>
00034 #include <vector>
00035
00036 /** \addtogroup Tracking
00037  * \{ */
00038 /** \addtogroup Tracking_gnuradio_blocks
00039  * \{ */
00040
00041
00042 class Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc;
00043
00044 using gps_l1_ca_dll_pll_tracking_gpu_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc>;
00045
00046 gps_l1_ca_dll_pll_tracking_gpu_cc_sptr
00047 gps_l1_ca_dll_pll_make_tracking_gpu_cc(
00048     int64_t fs_in,
00049     uint32_t vector_length,
00050     bool dump,
00051     std::string dump_filename,
00052     float pll_bw_hz,
00053     float dll_bw_hz,
00054     float early_late_space_chips);
00055
00056
00057 /*!
00058  * \brief This class implements a DLL + PLL tracking loop block
00059  */
00060 class Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc : public gr::block
00061 {
00062 public:
00063     ~Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc();
00064
00065     void set_channel(uint32_t channel);
00066     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
00067     void start_tracking();
00068
00069     int general_work(int noutput_items, gr_vector_int &ninput_items,
00070         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00071
00072     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00073
00074 private:
00075     friend gps_l1_ca_dll_pll_tracking_gpu_cc_sptr
00076     gps_l1_ca_dll_pll_make_tracking_gpu_cc(
00077         int64_t fs_in,
00078         uint32_t vector_length,
00079         bool dump,
00080         std::string dump_filename,
00081         float pll_bw_hz,
00082         float dll_bw_hz,
00083         float early_late_space_chips);
00084
00085     Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc(
00086         int64_t fs_in,
00087         uint32_t vector_length,
00088         bool dump,
00089         std::string dump_filename,
```

```
00090         float pll_bw_hz,
00091         float dll_bw_hz,
00092         float early_late_space_chips);
00093     void update_local_code();
00094     void update_local_carrier();
00095     void check_carrier_phase_coherent_initialization();
00096
00097     // PLL and DLL filter library
00098     Tracking_2nd_DLL_filter d_code_loop_filter;
00099     Tracking_FLL_PLL_filter d_carrier_loop_filter;
00100
00101     Gnss_Synchro *d_acquisition_gnss_synchro;
00102
00103     std::vector<gr_complex> d_Prompt_buffer;
00104
00105     // file dump
00106     std::string d_dump_filename;
00107     std::ofstream d_dump_file;
00108
00109     std::map<std::string, std::string> systemName;
00110     std::string sys;
00111
00112     // tracking configuration vars
00113     int64_t d_if_freq;
00114     int64_t d_fs_in;
00115     double d_early_late_spc_chips;
00116     uint32_t d_vector_length;
00117     uint32_t d_channel;
00118     int32_t d_n_correlator_taps;
00119
00120     // GPU HOST PINNED MEMORY IN/OUT VECTORS
00121     cuda_multicorrelator *multicorrelator_gpu;
00122     gr_complex *in_gpu;
00123     gr_complex *d_correlator_outs;
00124     gr_complex *d_ca_code;
00125     float *d_local_code_shift_chips;
00126
00127     gr_complex *d_Early;
00128     gr_complex *d_Prompt;
00129     gr_complex *d_Late;
00130
00131     // remaining code phase and carrier phase between tracking loops
00132     double d_rem_code_phase_samples;
00133     double d_rem_code_phase_chips;
00134     double d_rem_carrier_phase_rad;
00135
00136     // acquisition
00137     double d_acq_code_phase_samples;
00138     double d_acq_carrier_doppler_hz;
00139
00140     // tracking vars
00141     double d_code_freq_chips;
00142     double d_code_phase_step_chips;
00143     double d_carrier_doppler_hz;
00144     double d_carrier_phase_step_rad;
00145     double d_acc_carrier_phase_cycles;
00146     double d_code_phase_samples;
00147     double d_pll_to_dll_assist_secs_Ti;
00148
00149     // Integration period in samples
00150     int32_t d_correlation_length_samples;
00151
00152     // processing samples counters
00153     uint64_t d_sample_counter;
00154     uint64_t d_acq_sample_stamp;
00155
00156     // CN0 estimation and lock detector
00157     double d_carrier_lock_test;
00158     double d_CN0_SNV_dB_Hz;
00159     double d_carrier_lock_threshold;
00160     int32_t d_carrier_lock_fail_counter;
00161     int32_t d_cn0_estimation_counter;
00162
00163     // control vars
00164     bool d_acc_carrier_phase_initialized;
00165     bool d_enable_tracking;
00166     bool d_pull_in;
00167     bool d_dump;
00168 };
00169
00170
00171 /** \} */
00172 /** \} */
00173 #endif  // GNSS_SDR_GPS_L1_CA_DLL_PLL_TRACKING_GPU_CC_H
```

13.499 gps_l1_ca_gaussian_tracking_cc.h File Reference

Interface of a processing block of a DLL + Kalman carrier tracking loop for GPS L1 C/A signals.

```
#include "bayesian_estimation.h"
#include "cpu_multicorrelator_real_codes.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tracking_2nd_DLL_filter.h"
#include "tracking_2nd_PLL_filter.h"
#include <armadillo>
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

Classes

- class Gps_L1_Ca_Gaussian_Tracking_cc

  This class implements a DLL + PLL tracking loop block.

Typedefs

- using gps_l1_ca_gaussian_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Gaussian_Tracking_cc>

Functions

- gps_l1_ca_gaussian_tracking_cc_sptr **gps_l1_ca_gaussian_make_tracking_cc** (uint32_t order, int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float dll_bw_hz, float early_late_space_chips, bool bce_run, uint32_t bce_ptrans, uint32_t bce_strans, int32_t bce_nu, int32_t bce_kappa)

13.499.1 Detailed Description

Interface of a processing block of a DLL + Kalman carrier tracking loop for GPS L1 C/A signals.

Author

Javier Arribas, 2018. jarribas(at)cttc.es

Jordi Vila-Valls 2018. jvila(at)cttc.es

Carles Fernandez-Prades 2018. cfernandez(at)cttc.es

Reference: J. Vila-Valls, P. Closas, M. Navarro and C. Fernandez-Prades, "Are PLLs Dead? A Tutorial on Kalman Filter-based Techniques for Digital Carrier Synchronization", IEEE Aerospace and Electronic Systems Magazine, Vol. 32, No. 7, pp. 28–45, July 2017. DOI: 10.1109/MAES.2017.150260

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_l1_ca_gaussian_tracking_cc.h.

13.500 gps_l1_ca_gaussian_tracking_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_gaussian_tracking_cc.h
00003  * \brief Interface of a processing block of a DLL + Kalman carrier
00004  * tracking loop for GPS L1 C/A signals
00005  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00006  * \author Jordi Vila-Valls 2018. jvila(at)cttc.es
00007  * \author Carles Fernandez-Prades 2018. cfernandez(at)cttc.es
00008  *
```

```
00009  * Reference:
00010  * J. Vila-Valls, P. Closas, M. Navarro and C. Fernandez-Prades,
00011  * "Are PLLs Dead? A Tutorial on Kalman Filter-based Techniques for Digital
00012  * Carrier Synchronization", IEEE Aerospace and Electronic Systems Magazine,
00013  * Vol. 32, No. 7, pp. 28-45, July 2017. DOI: 10.1109/MAES.2017.150260
00014  *
00015  * -----------------------------------------------------------------------------
00016  *
00017  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00018  * This file is part of GNSS-SDR.
00019  *
00020  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00021  * SPDX-License-Identifier: GPL-3.0-or-later
00022  *
00023  * -----------------------------------------------------------------------------
00024  */
00025
00026 #ifndef GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_CC_H
00027 #define GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_CC_H
00028
00029 #if ARMA_NO_BOUND_CHECKING
00030 #define ARMA_NO_DEBUG 1
00031 #endif
00032
00033 #include "bayesian_estimation.h"
00034 #include "cpu_multicorrelator_real_codes.h"
00035 #include "gnss_block_interface.h"
00036 #include "gnss_synchro.h"
00037 #include "tracking_2nd_DLL_filter.h"
00038 #include "tracking_2nd_PLL_filter.h"
00039 #include <armadillo>
00040 #include <gnuradio/block.h>
00041 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00042 #include <fstream>
00043 #include <map>
00044 #include <string>
00045
00046 /** \addtogroup Tracking
00047  * \{ */
00048 /** \addtogroup Tracking_gnuradio_blocks
00049  * \{ */
00050
00051
00052 class Gps_L1_Ca_Gaussian_Tracking_cc;
00053
00054 using gps_l1_ca_gaussian_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Gaussian_Tracking_cc>;
00055
00056 gps_l1_ca_gaussian_tracking_cc_sptr
00057 gps_l1_ca_gaussian_make_tracking_cc(uint32_t order,
00058     int64_t fs_in,
00059     uint32_t vector_length,
00060     bool dump,
00061     const std::string& dump_filename,
00062     float dll_bw_hz,
00063     float early_late_space_chips,
00064     bool bce_run,
00065     uint32_t bce_ptrans,
00066     uint32_t bce_strans,
00067     int32_t bce_nu,
00068     int32_t bce_kappa);
00069
00070
00071 /*!
00072  * \brief This class implements a DLL + PLL tracking loop block
00073  */
00074 class Gps_L1_Ca_Gaussian_Tracking_cc : public gr::block
00075 {
00076 public:
00077     ~Gps_L1_Ca_Gaussian_Tracking_cc();
00078
00079     void set_channel(uint32_t channel);
00080     void set_gnss_synchro(Gnss_Synchro* p_gnss_synchro);
00081     void start_tracking();
00082
00083     int general_work(int noutput_items, gr_vector_int& ninput_items,
00084         gr_vector_const_void_star& input_items, gr_vector_void_star& output_items);
00085
00086     void forecast(int noutput_items, gr_vector_int& ninput_items_required);
00087
00088 private:
00089     friend gps_l1_ca_gaussian_tracking_cc_sptr
00090     gps_l1_ca_gaussian_make_tracking_cc(uint32_t order,
00091         int64_t fs_in,
00092         uint32_t vector_length,
00093         bool dump,
00094         const std::string& dump_filename,
00095         float dll_bw_hz,
```

```
00096         float early_late_space_chips,
00097         bool bce_run,
00098         uint32_t bce_ptrans,
00099         uint32_t bce_strans,
00100         int32_t bce_nu,
00101         int32_t bce_kappa);
00102
00103     Gps_L1_Ca_Gaussian_Tracking_cc(uint32_t order,
00104         int64_t fs_in,
00105         uint32_t vector_length,
00106         bool dump,
00107         const std::string& dump_filename,
00108         float dll_bw_hz,
00109         float early_late_space_chips,
00110         bool bce_run,
00111         uint32_t bce_ptrans,
00112         uint32_t bce_strans,
00113         int32_t bce_nu,
00114         int32_t bce_kappa);
00115
00116     int32_t save_matfile();
00117
00118     // tracking configuration vars
00119     uint32_t d_order;
00120     uint32_t d_vector_length;
00121     bool d_dump;
00122
00123     Gnss_Synchro* d_acquisition_gnss_synchro;
00124     uint32_t d_channel;
00125
00126     int64_t d_fs_in;
00127
00128     double d_early_late_spc_chips;
00129
00130     // remaining code phase and carrier phase between tracking loops
00131     double d_rem_code_phase_samples;
00132     double d_rem_code_phase_chips;
00133     float d_rem_carr_phase_rad;
00134
00135     // Kalman filter variables
00136     arma::mat kf_P_x_ini;  // initial state error covariance matrix
00137     arma::mat kf_P_x;      // state error covariance matrix
00138     arma::mat kf_P_x_pre;  // Predicted state error covariance matrix
00139     arma::mat kf_P_y;      // innovation covariance matrix
00140
00141     arma::mat kf_F;  // state transition matrix
00142     arma::mat kf_H;  // system matrix
00143     arma::mat kf_R;  // measurement error covariance matrix
00144     arma::mat kf_Q;  // system error covariance matrix
00145
00146     arma::colvec kf_x;      // state vector
00147     arma::colvec kf_x_pre;  // predicted state vector
00148     arma::colvec kf_y;      // measurement vector
00149     arma::mat kf_K;         // Kalman gain matrix
00150
00151     // Gaussian estimator
00152     Bayesian_estimator bayes_estimator;
00153     arma::mat kf_R_est;  // measurement error covariance
00154     uint32_t bayes_ptrans;
00155     uint32_t bayes_strans;
00156     int32_t bayes_nu;
00157     int32_t bayes_kappa;
00158
00159     bool bayes_run;
00160     uint32_t kf_iter;
00161
00162     // PLL and DLL filter library
00163     Tracking_2nd_DLL_filter d_code_loop_filter;
00164     // Tracking_2nd_PLL_filter d_carrier_loop_filter;
00165
00166     // acquisition
00167     double d_acq_carrier_doppler_step_hz{};
00168     double d_acq_code_phase_samples;
00169     double d_acq_carrier_doppler_hz;
00170     // correlator
00171     int32_t d_n_correlator_taps;
00172     volk_gnsssdr::vector<float> d_ca_code;
00173     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00174     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00175     Cpu_Multicorrelator_Real_Codes multicorrelator_cpu;
00176
00177     // tracking vars
00178     double d_code_freq_chips;
00179     double d_code_phase_step_chips;
00180     double d_code_phase_rate_step_chips;
00181     double d_carrier_doppler_hz;
00182     double d_carrier_dopplerrate_hz2;
```

```
00183     double d_carrier_phase_step_rad;
00184     double d_acc_carrier_phase_rad;
00185     double d_carr_phase_error_rad{};
00186     double d_carr_phase_sigma2;
00187     double d_code_phase_samples;
00188     double code_error_chips;
00189     double code_error_filt_chips;
00190
00191     // PRN period in samples
00192     int32_t d_current_prn_length_samples;
00193
00194     // processing samples counters
00195     uint64_t d_sample_counter;
00196     uint64_t d_acq_sample_stamp;
00197
00198     // CN0 estimation and lock detector
00199     int32_t d_cn0_estimation_counter;
00200     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00201     double d_carrier_lock_test;
00202     double d_CN0_SNV_dB_Hz;
00203     double d_carrier_lock_threshold;
00204     int32_t d_carrier_lock_fail_counter;
00205
00206     // control vars
00207     bool d_enable_tracking;
00208     bool d_pull_in;
00209
00210     // file dump
00211     std::string d_dump_filename;
00212     std::ofstream d_dump_file;
00213
00214     std::map<std::string, std::string> systemName;
00215     std::string sys;
00216 };
00217
00218
00219 /** \} */
00220 /** \} */
00221 #endif  // GNSS_SDR_GPS_L1_CA_GAUSSIAN_TRACKING_CC_H
```

13.501 gps_l1_ca_tcp_connector_tracking_cc.h File Reference

Interface of a TCP connector block based on code DLL + carrier PLL.

```
#include "cpu_multicorrelator.h"
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "tcp_communication.h"
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <fstream>
#include <map>
#include <string>
```

Classes

- class Gps_L1_Ca_Tcp_Connector_Tracking_cc

  This class implements a DLL + PLL tracking loop block.

Typedefs

- using gps_l1_ca_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Tcp_Connector_Tracking_cc>

Functions

- gps_l1_ca_tcp_connector_tracking_cc_sptr **gps_l1_ca_tcp_connector_make_tracking_cc** (int64_t fs_in, uint32_t vector_length, bool dump, const std::string &dump_filename, float early_late_space_chips, size_t port_ch0)

13.501.1 Detailed Description

Interface of a TCP connector block based on code DLL + carrier PLL.

Author

David Pubill, 2012. dpubill(at)cttc.es Javier Arribas, 2011. jarribas(at)cttc.es

Code DLL + carrier PLL according to the algorithms described in: K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_l1_ca_tcp_connector_tracking_cc.h.

13.502 gps_l1_ca_tcp_connector_tracking_cc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_l1_ca_tcp_connector_tracking_cc.h
00003  * \brief Interface of a TCP connector block based on code DLL + carrier PLL
00004  * \author David Pubill, 2012. dpubill(at)cttc.es
00005  *         Javier Arribas, 2011. jarribas(at)cttc.es
00006  *
00007  * Code DLL + carrier PLL according to the algorithms described in:
00008  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H.Jensen,
00009  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach,
00010  * Birkhauser, 2007
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_CC_H
00024 #define GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_CC_H
00025
00026 #include "cpu_multicorrelator.h"
00027 #include "gnss_block_interface.h"
00028 #include "gnss_synchro.h"
00029 #include "tcp_communication.h"
00030 #include <gnuradio/block.h>
00031 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00032 #include <fstream>
00033 #include <map>
00034 #include <string>
00035
00036 /** \addtogroup Tracking
00037  * \{ */
00038 /** \addtogroup Tracking_gnuradio_blocks
00039  * \{ */
00040
00041
00042 class Gps_L1_Ca_Tcp_Connector_Tracking_cc;
00043
00044 using gps_l1_ca_tcp_connector_tracking_cc_sptr = gnss_shared_ptr<Gps_L1_Ca_Tcp_Connector_Tracking_cc>;
00045
00046 gps_l1_ca_tcp_connector_tracking_cc_sptr
00047 gps_l1_ca_tcp_connector_make_tracking_cc(
00048     int64_t fs_in, uint32_t vector_length,
00049     bool dump,
00050     const std::string &dump_filename,
00051     float early_late_space_chips,
00052     size_t port_ch0);
00053
00054
00055 /*!
00056  * \brief This class implements a DLL + PLL tracking loop block
00057  */
00058 class Gps_L1_Ca_Tcp_Connector_Tracking_cc : public gr::block
00059 {
00060 public:
00061     ~Gps_L1_Ca_Tcp_Connector_Tracking_cc();
00062
00063     void set_channel(uint32_t channel);
00064     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
```

```
00065     void start_tracking();
00066
00067     /*
00068      * \brief just like gr_block::general_work, only this arranges to call consume_each for you
00069      *
00070      * The user must override work to define the signal processing code
00071      */
00072     int general_work(int noutput_items, gr_vector_int &ninput_items,
00073         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00074
00075     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00076
00077 private:
00078     friend gps_l1_ca_tcp_connector_tracking_cc_sptr
00079     gps_l1_ca_tcp_connector_make_tracking_cc(
00080         int64_t fs_in, uint32_t vector_length,
00081         bool dump,
00082         const std::string &dump_filename,
00083         float early_late_space_chips,
00084         size_t port_ch0);
00085
00086     Gps_L1_Ca_Tcp_Connector_Tracking_cc(
00087         int64_t fs_in, uint32_t vector_length,
00088         bool dump,
00089         const std::string &dump_filename,
00090         float early_late_space_chips,
00091         size_t port_ch0);
00092
00093     volk_gnsssdr::vector<gr_complex> d_ca_code;
00094     // correlator
00095     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00096     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00097     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00098     Cpu_Multicorrelator multicorrelator_cpu;
00099     Tcp_Communication d_tcp_com;
00100     Gnss_Synchro *d_acquisition_gnss_synchro;
00101     // tracking configuration vars
00102
00103     gr_complex *d_Early;
00104     gr_complex *d_Prompt;
00105     gr_complex *d_Late;
00106
00107     // file dump
00108     std::string d_dump_filename;
00109     std::ofstream d_dump_file;
00110
00111     std::map<std::string, std::string> systemName;
00112     std::string sys;
00113
00114     double d_early_late_spc_chips;
00115     double d_code_phase_step_chips;
00116     double d_rem_code_phase_samples;
00117     double d_next_rem_code_phase_samples;
00118     double d_code_freq_hz;
00119     double d_carrier_doppler_hz;
00120     double d_acc_carrier_phase_rad;
00121     double d_code_phase_samples;
00122     double d_sample_counter_seconds;
00123
00124     int64_t d_fs_in;
00125     uint64_t d_sample_counter;
00126     uint64_t d_acq_sample_stamp;
00127
00128     size_t d_port_ch0;
00129     size_t d_port;
00130
00131     uint32_t d_vector_length;
00132     uint32_t d_channel;
00133
00134     int32_t d_correlation_length_samples;
00135     int32_t d_n_correlator_taps;
00136     int32_t d_listen_connection;
00137     int32_t d_current_prn_length_samples;
00138     int32_t d_next_prn_length_samples;
00139     int32_t d_cn0_estimation_counter;
00140     int32_t d_carrier_lock_fail_counter;
00141
00142     float d_rem_carr_phase_rad;
00143     float d_acq_code_phase_samples;
00144     float d_acq_carrier_doppler_hz;
00145     float d_carrier_lock_test;
00146     float d_CN0_SNV_dB_Hz;
00147     float d_carrier_lock_threshold;
00148     float d_control_id;
00149
00150     bool d_enable_tracking;
00151     bool d_pull_in;
```

```
00152     bool d_dump;
00153 };
00154
00155
00156 /** \} */
00157 /** \} */
00158 #endif  // GNSS_SDR_GPS_L1_CA_TCP_CONNECTOR_TRACKING_CC_H
```

13.503 kf_tracking.h

```
00001 /*!
00002  * \file kf_tracking.cc
00003  * \brief Implementation of a Kalman filter based tracking with optional Vector
00004  * Tracking Loop message receiver block.
00005  * \author Javier Arribas, 2020. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00010  *
00011  * GNSS-SDR is a software defined Global Navigation
00012  *          Satellite Systems receiver
00013  *
00014  * This file is part of GNSS-SDR.
00015  *
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_KF_TRACKING_H
00022 #define GNSS_SDR_KF_TRACKING_H
00023
00024 #if ARMA_NO_BOUND_CHECKING
00025 #define ARMA_NO_DEBUG 1
00026 #endif
00027
00028 #include "cpu_multicorrelator_real_codes.h"
00029 #include "exponential_smoother.h"
00030 #include "gnss_block_interface.h"
00031 #include "gnss_time.h"  // for timetags produced by File_Timestamp_Signal_Source
00032 #include "kf_conf.h"
00033 #include "tracking_FLL_PLL_filter.h"  // for PLL/FLL filter
00034 #include "tracking_loop_filter.h"     // for DLL filter
00035 #include <armadillo>
00036 #include <boost/circular_buffer.hpp>
00037 #include <gnuradio/block.h>                  // for block
00038 #include <gnuradio/gr_complex.h>             // for gr_complex
00039 #include <gnuradio/types.h>                  // for gr_vector_int, gr_vector...
00040 #include <pmt/pmt.h>                         // for pmt_t
00041 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>  // for volk_gnsssdr::vector
00042 #include <cstddef>                           // for size_t
00043 #include <cstdint>                           // for int32_t
00044 #include <fstream>                           // for ofstream
00045 #include <memory>
00046 #include <string>    // for string
00047 #include <typeinfo>  // for typeid
00048 #include <utility>   // for pair
00049
00050 class Gnss_Synchro;
00051 class kf_tracking;
00052
00053 using kf_tracking_sptr = gnss_shared_ptr<kf_tracking>;
00054
00055 kf_tracking_sptr kf_make_tracking(const Kf_Conf &conf_);
00056
00057 /*!
00058  * \brief This class implements a code DLL + carrier PLL tracking block.
00059  */
00060 class kf_tracking : public gr::block
00061 {
00062 public:
00063     ~kf_tracking();
00064
00065     void set_channel(uint32_t channel);
00066     void set_gnss_synchro(Gnss_Synchro *p_gnss_synchro);
00067     void start_tracking();
00068     void stop_tracking();
00069
00070     int general_work(int noutput_items, gr_vector_int &ninput_items,
00071         gr_vector_const_void_star &input_items, gr_vector_void_star &output_items);
00072
00073     void forecast(int noutput_items, gr_vector_int &ninput_items_required);
00074
```

```
00075 private:
00076     friend kf_tracking_sptr kf_make_tracking(const Kf_Conf &conf_);
00077     explicit kf_tracking(const Kf_Conf &conf_);
00078
00079     void init_kf(double acq_code_phase_chips, double acq_doppler_hz);
00080     void update_kf_narrow_integration_time();
00081     void update_kf_cn0(double current_cn0_dbhz);
00082     void run_Kf();
00083
00084     void msg_handler_telemetry_to_trk(const pmt::pmt_t &msg);
00085     void msg_handler_pvt_to_trk(const pmt::pmt_t &msg);
00086     void do_correlation_step(const gr_complex *input_samples);
00087
00088     void check_carrier_phase_coherent_initialization();
00089     void update_tracking_vars();
00090     void clear_tracking_vars();
00091     void save_correlation_results();
00092     void log_data();
00093     bool cn0_and_tracking_lock_status(double coh_integration_time_s);
00094     bool acquire_secondary();
00095     int32_t save_matfile() const;
00096
00097     Cpu_Multicorrelator_Real_Codes d_multicorrelator_cpu;
00098
00099     Kf_Conf d_trk_parameters;
00100
00101     Exponential_Smoother d_cn0_smoother;
00102     Exponential_Smoother d_carrier_lock_test_smoother;
00103
00104     Gnss_Synchro *d_acquisition_gnss_synchro;
00105
00106     volk_gnsssdr::vector<float> d_tracking_code;
00107     volk_gnsssdr::vector<float> d_data_code;
00108     volk_gnsssdr::vector<float> d_local_code_shift_chips;
00109     volk_gnsssdr::vector<gr_complex> d_correlator_outs;
00110     volk_gnsssdr::vector<gr_complex> d_Prompt_Data;
00111     volk_gnsssdr::vector<gr_complex> d_Prompt_buffer;
00112
00113     boost::circular_buffer<gr_complex> d_Prompt_circular_buffer;
00114     boost::circular_buffer<std::pair<double, double» d_code_ph_history;
00115     boost::circular_buffer<std::pair<double, double» d_carr_ph_history;
00116
00117     const size_t d_int_type_hash_code = typeid(int).hash_code();
00118
00119     // Kalman Filter class variables
00120     arma::mat d_F;
00121     arma::mat d_H;
00122     arma::mat d_R;
00123     arma::mat d_Q;
00124     arma::mat d_P_old_old;
00125     arma::mat d_P_new_old;
00126     arma::mat d_P_new_new;
00127     arma::vec d_x_old_old;
00128     arma::vec d_x_new_old;
00129     arma::vec d_x_new_new;
00130
00131     std::string d_secondary_code_string;
00132     std::string d_data_secondary_code_string;
00133     std::string d_systemName;
00134     std::string d_signal_type;
00135     std::string d_signal_pretty_name;
00136     std::string d_dump_filename;
00137
00138     std::ofstream d_dump_file;
00139
00140     gr_complex *d_Very_Early;
00141     gr_complex *d_Early;
00142     gr_complex *d_Prompt;
00143     gr_complex *d_Late;
00144     gr_complex *d_Very_Late;
00145
00146     gr_complex d_VE_accu;
00147     gr_complex d_E_accu;
00148     gr_complex d_P_accu;
00149     gr_complex d_P_accu_old;
00150     gr_complex d_L_accu;
00151     gr_complex d_VL_accu;
00152     gr_complex d_P_data_accu;
00153
00154     // nominal signal parameters
00155     double d_signal_carrier_freq;
00156     double d_code_period;
00157     double d_code_chip_rate;
00158
00159     // acquisition
00160     double d_acq_code_phase_samples;
00161     double d_acq_carrier_doppler_hz;
```

```
00162     double d_current_correlation_time_s;
00163
00164     // carrier and code discriminators output
00165     double d_carr_phase_error_disc_hz;
00166     double d_code_error_disc_chips;
00167
00168     // estimated parameters
00169     // code
00170     double d_code_error_kf_chips;
00171     double d_code_freq_kf_chips_s;
00172     // carrier
00173     double d_carrier_phase_kf_rad;
00174     double d_carrier_doppler_kf_hz;
00175     double d_carrier_doppler_rate_kf_hz_s;
00176
00177     double d_acc_carrier_phase_rad;
00178
00179     double d_T_chip_seconds;
00180     double d_T_prn_seconds;
00181     double d_T_prn_samples;
00182     double d_K_blk_samples;
00183     double d_carrier_lock_test;
00184     double d_CN0_SNV_dB_Hz;
00185     double d_carrier_lock_threshold;
00186
00187     // carrier NCO
00188     double d_carrier_phase_step_rad;
00189     double d_carrier_phase_rate_step_rad;
00190
00191     // code NCO
00192     double d_code_phase_step_chips;
00193     double d_code_phase_rate_step_chips;
00194     double d_rem_code_phase_chips;
00195     double d_rem_code_phase_samples;
00196
00197     double d_beta;
00198
00199     uint64_t d_sample_counter;
00200     uint64_t d_acq_sample_stamp;
00201
00202     float *d_prompt_data_shift;
00203     float d_rem_carr_phase_rad;
00204
00205     uint32_t d_channel;
00206     uint32_t d_secondary_code_length;
00207     uint32_t d_data_secondary_code_length;
00208
00209     int32_t d_symbols_per_bit;
00210     int32_t d_state;
00211     int32_t d_correlation_length_ms;
00212     int32_t d_n_correlator_taps;
00213     int32_t d_current_prn_length_samples;
00214     int32_t d_extend_correlation_symbols_count;
00215     int32_t d_current_symbol;
00216     int32_t d_current_data_symbol;
00217     int32_t d_cn0_estimation_counter;
00218     int32_t d_carrier_lock_fail_counter;
00219     int32_t d_code_lock_fail_counter;
00220     int32_t d_code_samples_per_chip;  // All signals have 1 sample per chip code except Gal. E1 which
      has 2 (CBOC disabled) or 12 (CBOC enabled)
00221     int32_t d_code_length_chips;
00222
00223     bool d_pull_in_transitory;
00224     bool d_corrected_doppler;
00225     bool d_interchange_iq;
00226     bool d_veml;
00227     bool d_cloop;
00228     bool d_secondary;
00229     bool d_dump;
00230     bool d_dump_mat;
00231     bool d_acc_carrier_phase_initialized;
00232     bool d_carrier_phase_discontinuity{true};  // pending report of a new carrier phase ambiguity
00233     bool d_enable_extended_integration;
00234 };
00235
00236 #endif  // GNSS_SDR_KF_TRACKING_H
```

13.504 bayesian_estimation.h File Reference

Interface of a library with Bayesian noise statistic estimation.

```
#include <armadillo>
#include <gnuradio/gr_complex.h>
```

Classes

- class Bayesian_estimator

  Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance).

13.504.1 Detailed Description

Interface of a library with Bayesian noise statistic estimation.
Bayesian_estimator is a Bayesian estimator which attempts to estimate the properties of a stochastic process based on a sequence of discrete samples of the sequence.
[1]: LaMountain, Gerald, Vilà-Valls, Jordi, Closas, Pau, "Bayesian Covariance Estimation for Kalman Filter based Digital Carrier Synchronization," Proceedings of the 31st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2018), Miami, Florida, September 2018, pp. 3575-3586. https://doi.org/10.33012/2018.15911

Authors

- Gerald LaMountain, 2018. gerald(at)ece.neu.edu
- Jordi Vila-Valls 2018. jvila(at)cttc.es

13.504.1.1

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file bayesian_estimation.h.

13.505 bayesian_estimation.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file bayesian_estimation.h
00003  * \brief Interface of a library with Bayesian noise statistic estimation
00004  *
00005  * Bayesian_estimator is a Bayesian estimator which attempts to estimate
00006  * the properties of a stochastic process based on a sequence of
00007  * discrete samples of the sequence.
00008  *
00009  * [1]: LaMountain, Gerald, Vilà-Valls, Jordi, Closas, Pau, "Bayesian
00010  * Covariance Estimation for Kalman Filter based Digital Carrier
00011  * Synchronization," Proceedings of the 31st International Technical Meeting
00012  * of the Satellite Division of The Institute of Navigation
00013  * (ION GNSS+ 2018), Miami, Florida, September 2018, pp. 3575-3586.
00014  * https://doi.org/10.33012/2018.15911
00015  *
00016  * \authors <ul>
00017  *          <li> Gerald LaMountain, 2018. gerald(at)ece.neu.edu
00018  *          <li> Jordi Vila-Valls 2018. jvila(at)cttc.es
00019  *          </ul>
00020  * -----------------------------------------------------------------------------
00021  *
00022  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00023  * This file is part of GNSS-SDR.
00024  *
00025  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00026  * SPDX-License-Identifier: GPL-3.0-or-later
00027  *
00028  * -----------------------------------------------------------------------------
00029  */
00030
00031 #ifndef GNSS_SDR_BAYESIAN_ESTIMATION_H
00032 #define GNSS_SDR_BAYESIAN_ESTIMATION_H
00033
00034 #if ARMA_NO_BOUND_CHECKING
00035 #define ARMA_NO_DEBUG 1
00036 #endif
00037
00038 #include <armadillo>
00039 #include <gnuradio/gr_complex.h>
00040
00041 /** \addtogroup Tracking
00042  * \{ */
00043 /** \addtogroup Tracking_libs
00044  * \{ */
```

```
00045
00046
00047 /*! \brief Bayesian_estimator is an estimator of noise characteristics (i.e. mean, covariance)
00048  *
00049  * Bayesian_estimator is an estimator which performs estimation of noise characteristics from
00050  * a sequence of identically and independently distributed (IID) samples of a stationary
00051  * stochastic process by way of Bayesian inference using conjugate priors. The posterior
00052  * distribution is assumed to be Gaussian with mean \mathbf{\mu} and covariance \hat{\mathbf{C}},
00053  * which has a conjugate prior given by a normal-inverse-Wishart distribution with parameters
00054  * \mathbf{\mu}_{0}, \kappa_{0}, \nu_{0}, and \mathbf{\Psi}.
00055  *
00056  * [1] TODO: Ref1
00057  *
00058  */
00059
00060 class Bayesian_estimator
00061 {
00062 public:
00063     Bayesian_estimator();
00064     explicit Bayesian_estimator(int ny);
00065     Bayesian_estimator(const arma::vec& mu_prior_0, int kappa_prior_0, int nu_prior_0, const
      arma::mat& Psi_prior_0);
00066     ~Bayesian_estimator() = default;
00067
00068     void init(const arma::mat& mu_prior_0, int kappa_prior_0, int nu_prior_0, const arma::mat&
      Psi_prior_0);
00069
00070     void update_sequential(const arma::vec& data);
00071     void update_sequential(const arma::vec& data, const arma::vec& mu_prior_0, int kappa_prior_0, int
      nu_prior_0, const arma::mat& Psi_prior_0);
00072
00073     arma::mat get_mu_est() const;
00074     arma::mat get_Psi_est() const;
00075
00076 private:
00077     arma::vec mu_est;
00078     arma::mat Psi_est;
00079     arma::vec mu_prior;
00080     arma::mat Psi_prior;
00081     int kappa_prior;
00082     int nu_prior;
00083 };
00084
00085
00086 /** \} */
00087 /** \} */
00088 #endif  // GNSS_SDR_BAYESIAN_ESTIMATION_H
```

13.506 bit_synchronizer.h File Reference

Histogram-based bit-edge synchronizer for GNSS prompt correlator outputs.

```
#include <cmath>
#include <complex>
#include <cstdint>
#include <deque>
#include <vector>
```

Classes

- class HistogramBitSynchronizer

  Histogram-based navigation data bit synchronizer.
- struct HistogramBitSynchronizer::Config

  Configuration parameters for HistogramBitSynchronizer.

13.506.1 Detailed Description

Histogram-based bit-edge synchronizer for GNSS prompt correlator outputs.

Author

Carles Fernandez-Prades, 2026 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file bit_synchronizer.h.

13.507 bit_synchronizer.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file bit_synchronizer.h
00003  * \brief Histogram-based bit-edge synchronizer for GNSS prompt correlator outputs.
00004  * \author Carles Fernandez-Prades, 2026 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BIT_SYNCHRONIZER_H
00018 #define GNSS_SDR_BIT_SYNCHRONIZER_H
00019
00020 #include <cmath>
00021 #include <complex>
00022 #include <cstdint>
00023 #include <deque>
00024 #include <vector>
00025
00026 /** \addtogroup Tracking
00027  * \{ */
00028 /** \addtogroup Tracking_libs
00029  * \{ */
00030
00031
00032 /**
00033  * @brief Histogram-based navigation data bit synchronizer.
00034  */
00035 class HistogramBitSynchronizer
00036 {
00037 public:
00038     /**
00039      * @brief Configuration parameters for HistogramBitSynchronizer.
00040      *
00041      * These parameters define the bit period, the update cadence, the lock criteria,
00042      * and the transition detection method.
00043      */
00044     struct Config
00045     {
00046         /**
00047          * @brief Navigation data bit period in milliseconds.
00048          *
00049          * This is the nominal duration of one navigation data bit.
00050          */
00051         int bit_period_ms;
00052
00053         /**
00054          * @brief Time interval between successive calls to update(), in milliseconds.
00055          *
00056          * This should match the minimum integration interval (epoch) produced by the
00057          * tracking loop and used to generate the provided prompt correlator output.
00058          */
00059         int epoch_ms;
00060
00061         /**
00062          * @brief Minimum number of detected transition events required before lock evaluation.
00063          *
00064          * The histogram is built from detected candidate transitions. Lock decisions are
00065          * not attempted until at least this many events have been accumulated.
00066          *
00067          * Trade-offs:
00068          *   - Larger values increase robustness against false locks but increase time-to-lock.
00069          *   - Smaller values reduce time-to-lock but increase sensitivity to noise/spurious
      transitions.
00070          */
00071         int min_events_for_lock;
```

```
00072
00073         /**
00074          * @brief Required dominance ratio of the winning histogram bin.
00075          *
00076          * Lock requires the most frequent histogram bin to be sufficiently dominant:
00077          * @f[
00078          *   \text{dominance\_ratio} = \frac{\text{best\_bin\_count}}{\text{total\_detected\_events}}
00079          * @f]
00080          *
00081          * Guidance:
00082          *   - Values near 0.5 may lock faster but increase false-lock probability.
00083          *   - Values closer to 1.0 are conservative and require a clearly dominant phase.
00084          */
00085         double dominance_ratio;
00086
00087         /**
00088          * @brief Required normalized margin between the best and second-best bins.
00089          *
00090          * The margin is computed as (best_count - second_best_count) / total_events.
00091          * This prevents lock when two candidate phases have similar support.
00092          */
00093         double runner_up_margin;
00094
00095         /**
00096          * @brief Required stability of the dominant histogram bin (consecutive evaluations).
00097          *
00098          * Even if the dominance ratio is met, the algorithm requires that the same histogram
00099          * bin remains dominant for this many consecutive event-driven lock evaluations before
00100          * declaring lock. Evaluations without a newly detected transition do not count toward
00101          * this requirement.
00102          *
00103          * This helps prevent locking on transient peaks caused by noise or short-lived disturbances.
00104          */
00105         int stable_best_required;
00106
00107         /**
00108          * @brief Minimum magnitude of the prompt correlator output.
00109          *
00110          * Candidate transition detection is suppressed when @f$|P| < \text{min\_prompt\_mag}@f$,
00111          * where @f$P@f$ is the prompt correlator output.
00112          *
00113          * Use this to avoid counting unreliable transitions when tracking quality is poor or
00114          * the prompt output is dominated by noise.
00115          */
00116         float min_prompt_mag;
00117
00118         /**
00119          * @brief Number of prompt epochs averaged on each side of a candidate edge.
00120          *
00121          * A transition is evaluated after this many prompt epochs have been observed on
00122          * both sides of the candidate boundary. Values larger than one improve rejection
00123          * of isolated noisy prompt inversions at the cost of a short detection delay.
00124          */
00125         int transition_window_epochs;
00126
00127         /**
00128          * @brief Minimum normalized transition confidence required for a histogram vote.
00129          *
00130          * Confidence combines the phase inversion between the pre-edge and post-edge
00131          * prompt averages with their internal coherence. It lies in the range [-1, 1],
00132          * with values near 1 indicating a clean polarity inversion.
00133          */
00134         double transition_confidence;
00135
00136         /**
00137          * @brief Number of fresh phase-matching events required after tentative lock.
00138          *
00139          * Once the histogram satisfies its dominance, runner-up margin, and stability
00140          * requirements, this many subsequent transition events must occur in the same
00141          * phase bin before lock is declared. Winner stability and tentative validation
00142          * advance concurrently; they are not applied as serial waiting periods.
00143          */
00144         int tentative_events_required;
00145
00146         /**
00147          * @brief Select the transition detection method.
00148          *
00149          * If true (recommended), compares coherent prompt averages immediately before
00150          * and after each candidate boundary. A transition is detected when:
00151          *     @f[
00152          *       -\frac{\Re\{ A_{after} A^*_{before} \}}
00153          *       {|A_{after}| |A_{before}|}
00154          *     @f]
00155          * combined with the internal coherence of each averaging window exceeds
00156          * Config::transition_confidence.
00157          *
00158          * This method is insensitive to constant carrier phase rotations and rejects isolated
```

```
00159          * prompt inversions and ambiguous samples near a candidate boundary.
00160          *
00161          * If false, uses a simpler sign-change detector on the real part:
00162          *   - A candidate transition is detected when sign(Re(P_k)) != sign(Re(P_{k-1})).
00163          *
00164          * This assumes the prompt output is already aligned with the data bit polarity
00165          * (i.e., stable PLL lock and correct navigation bit polarity mapping).
00166          */
00167         bool use_phase_dot_detector;
00168
00169         Config()
00170             : bit_period_ms(20),
00171               epoch_ms(1),
00172               min_events_for_lock(6),
00173               dominance_ratio(0.5),
00174               runner_up_margin(0.3),
00175               stable_best_required(3),
00176               min_prompt_mag(0.0f),
00177               transition_window_epochs(4),
00178               transition_confidence(0.6),
00179               tentative_events_required(2),
00180               use_phase_dot_detector(true)
00181         {
00182         }
00183     };
00184
00185     /**
00186      * @brief Construct a histogram bit synchronizer with the provided configuration.
00187      *
00188      * Initializes internal counters and allocates the histogram with bins() entries,
00189      * all set to zero.
00190      *
00191      * @param cfg Configuration parameters.
00192      */
00193     explicit HistogramBitSynchronizer(const Config& cfg)
00194         : cfg_(cfg),
00195           hist_(),
00196           total_events_(0),
00197           epoch_count_(0),
00198           edge_phase_(-1),
00199           last_sign_(+1),
00200           last_best_bin_(0),
00201           stable_best_count_(0),
00202           tentative_phase_(-1),
00203           tentative_event_count_(0),
00204           locked_(false),
00205           has_last_sign_(false),
00206           has_last_best_bin_(false),
00207           tentative_lock_(false)
00208     {
00209         hist_.assign(bins(), 0);
00210     }
00211
00212     /**
00213      * @brief Reset the synchronizer state.
00214      *
00215      * Clears the histogram and all internal counters/flags, returning the instance to the
00216      * pre-lock state:
00217      *   - locked() becomes false
00218      *   - edge_phase() becomes -1
00219      *   - total_events and epoch_count are reset to zero
00220      */
00221     void reset();
00222
00223     /**
00224      * @brief Update the synchronizer once per epoch.
00225      *
00226      * This method should be called at a fixed cadence defined by Config::epoch_ms
00227      *
00228      * The method:
00229      *   - Advances the internal epoch counter,
00230      *   - Optionally performs candidate transition detection if tracking quality is acceptable,
00231      *   - Updates the phase histogram on detected transitions,
00232      *   - Evaluates lock once enough events have been gathered.
00233      *
00234      * @param prompt Prompt correlator output for the current epoch.
00235      * @param tracking_quality_ok Indicates whether tracking quality is sufficient to trust
00236      *        the prompt sample for transition detection (e.g., code/carrier lock metrics).
00237      *
00238      * @return True only on the epoch when lock is first declared; false otherwise
00239      *         (including subsequent epochs after lock has been achieved).
00240      */
00241     bool update(const std::complex<float>& prompt, bool tracking_quality_ok);
00242
00243     /**
00244      * @brief Query whether the synchronizer has achieved lock.
00245      *
```

```
00246      * @return True if lock has been declared, false otherwise.
00247      */
00248     bool locked() const { return locked_; }
00249
00250     /**
00251      * @brief Get the estimated bit edge phase bin.
00252      *
00253      * The edge phase is expressed as an integer histogram bin index in the range
00254      * [0, bins()-1] when locked. The interpretation is "which epoch phase within the
00255      * bit period is most likely to contain a navigation bit transition."
00256      *
00257      * @return Estimated edge phase bin index, or -1 if not locked.
00258      */
00259     int edge_phase() const { return edge_phase_; }
00260
00261     /**
00262      * @brief Predict whether a given epoch index corresponds to a bit edge.
00263      *
00264      * For a given epoch index @p k (0-based), this function returns true when @p k is
00265      * aligned with the currently estimated edge phase (i.e., the predicted transition epoch),
00266      * and false otherwise.
00267      *
00268      * If not locked, this always returns false.
00269      *
00270      * @param k Epoch index (0-based, consistent with the caller's epoch counting).
00271      * @return True if @p k is the predicted edge epoch; false otherwise.
00272      */
00273     bool is_edge_epoch(std::int64_t k) const;
00274
00275     /**
00276      * @brief Return the number of histogram bins.
00277      *
00278      * Derived from the bit period and epoch duration, e.g.:
00279      *   bins = bit_period_ms / epoch_ms
00280      *
00281      * @return Number of histogram bins.
00282      */
00283     int bins() const;
00284
00285     /**
00286      * @brief Access the internal histogram (read-only).
00287      *
00288      * Each entry counts how many detected candidate transitions occurred at the
00289      * corresponding phase bin within the bit period.
00290      *
00291      * @return Reference to the histogram vector.
00292      */
00293     const std::vector<int>& get_histogram() const { return hist_; }
00294
00295     /**
00296      * @brief Total number of detected transition events accumulated into the histogram.
00297      *
00298      * @return Total detected events.
00299      */
00300     std::int64_t get_total_events() const { return total_events_; }
00301
00302     /**
00303      * @brief Total number of epochs processed by update().
00304      *
00305      * This counter increments once per call to update(), regardless of whether a transition
00306      * is detected or whether tracking_quality_ok is true.
00307      *
00308      * @return Total processed epochs.
00309      */
00310     std::int64_t get_epoch_count() const { return epoch_count_; }
00311
00312     /**
00313      * @brief Return the number of epochs until the next predicted navigation bit edge.
00314      *
00315      * When the synchronizer is locked, this function computes the forward distance
00316      * (in epochs) from the most recently processed epoch to the next epoch that is
00317      * aligned with the estimated bit-edge phase.
00318      *
00319      * The result is expressed modulo the bit period and has the following meaning:
00320      *   - 0  : the current epoch corresponds to the predicted start of a new navigation bit
00321      *   - >0 : number of epochs remaining until the next bit boundary
00322      *
00323      * The computation is based on the internal epoch counter advanced by update(),
00324      * assuming that update() is called once per epoch with a constant cadence
00325      * equal to Config::epoch_ms.
00326      *
00327      * If the synchronizer is not locked, or if the configuration yields an invalid
00328      * number of bins, this function returns -1.
00329      *
00330      * @return Number of epochs until the next predicted bit edge, or -1 if not locked
00331      *         or if the bit period configuration is invalid.
00332      */
```

```
00333     int epochs_until_next_edge() const;
00334
00335 private:
00336     void best_bins_and_counts(int& best_bin, int& best_count, int& second_best_count) const;
00337     bool histogram_confident(int best_count, int second_best_count) const;
00338     int transition_window_epochs() const;
00339
00340     Config cfg_;
00341     std::vector<int> hist_;
00342     std::deque<std::complex<float» prompt_window_;
00343
00344     std::int64_t total_events_;
00345     std::int64_t epoch_count_;
00346
00347     int edge_phase_;
00348     int last_sign_;          // Sign history (for simple detector)
00349     int last_best_bin_;      // Stability tracking for best bin
00350     int stable_best_count_;  // Stability tracking for best bin
00351     int tentative_phase_;
00352     int tentative_event_count_;
00353
00354     bool locked_;
00355     bool has_last_sign_;      // Sign history (for simple detector)
00356     bool has_last_best_bin_;  // Stability tracking for best bin
00357     bool tentative_lock_;
00358 };
00359
00360 /** \} */
00361 /** \} */
00362 #endif  // GNSS_SDR_BIT_SYNCHRONIZER_H
```

13.508 cpu_multicorrelator.h File Reference

High optimized CPU vector multiTAP correlator class.

`#include <complex>`

Classes

- class Cpu_Multicorrelator

  Class that implements carrier wipe-off and correlators.

13.508.1 Detailed Description

High optimized CPU vector multiTAP correlator class.

Authors

- Javier Arribas, 2015. jarribas(at)cttc.es

Class that implements a high optimized vector multiTAP correlator class for CPUs
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cpu_multicorrelator.h.

13.509 cpu_multicorrelator.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cpu_multicorrelator.h
00003  * \brief High optimized CPU vector multiTAP correlator class
00004  * \authors <ul>
00005  *          <li> Javier Arribas, 2015. jarribas(at)cttc.es
00006  *          </ul>
00007  *
00008  * Class that implements a high optimized vector multiTAP correlator class for CPUs
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
```

```
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CPU_MULTICORRELATOR_H
00022 #define GNSS_SDR_CPU_MULTICORRELATOR_H
00023
00024
00025 #include <complex>
00026
00027 /** \addtogroup Tracking
00028  * \{ */
00029 /** \addtogroup Tracking_libs tracking_libs
00030  * Utitilies for GNSS signal tracking.
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief Class that implements carrier wipe-off and correlators.
00036  */
00037 class Cpu_Multicorrelator
00038 {
00039 public:
00040     Cpu_Multicorrelator() = default;
00041     ~Cpu_Multicorrelator();
00042     bool init(int max_signal_length_samples, int n_correlators);
00043     bool set_local_code_and_taps(int code_length_chips, const std::complex<float> *local_code_in,
      float *shifts_chips);
00044     bool set_input_output_vectors(std::complex<float> *corr_out, const std::complex<float> *sig_in);
00045     void update_local_code(int correlator_length_samples, float rem_code_phase_chips, float
      code_phase_step_chips);
00046     bool Carrier_wipeoff_multicorrelator_resampler(float rem_carrier_phase_in_rad, float
      phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, int signal_length_samples);
00047     bool free();
00048
00049 private:
00050     // Allocate the device input vectors
00051     const std::complex<float> *d_sig_in{nullptr};
00052     const std::complex<float> *d_local_code_in{nullptr};
00053     std::complex<float> **d_local_codes_resampled{nullptr};
00054     std::complex<float> *d_corr_out{nullptr};
00055     float *d_shifts_chips{nullptr};
00056     int d_code_length_chips{0};
00057     int d_n_correlators{0};
00058 };
00059
00060
00061 /** \} */
00062 /** \} */
00063 #endif  // GNSS_SDR_CPU_MULTICORRELATOR_H
```

13.510 cpu_multicorrelator_16sc.h File Reference

Highly optimized CPU vector multiTAP correlator class for lv_16sc_t (short int complex).

`#include <volk_gnsssdr/volk_gnsssdr.h>`

Classes

- class Cpu_Multicorrelator_16sc

  Class that implements carrier wipe-off and correlators.

13.510.1 Detailed Description

Highly optimized CPU vector multiTAP correlator class for lv_16sc_t (short int complex).

Authors

- Javier Arribas, 2016. jarribas(at)cttc.es

Class that implements a highly optimized vector multiTAP correlator class for CPUs
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cpu_multicorrelator_16sc.h.

13.511 cpu_multicorrelator_16sc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cpu_multicorrelator_16sc.h
00003  * \brief Highly optimized CPU vector multiTAP correlator class for lv_16sc_t (short int complex)
00004  * \authors <ul>
00005  *          <li> Javier Arribas, 2016. jarribas(at)cttc.es
00006  *          </ul>
00007  *
00008  * Class that implements a highly optimized vector multiTAP correlator class for CPUs
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CPU_MULTICORRELATOR_16SC_H
00022 #define GNSS_SDR_CPU_MULTICORRELATOR_16SC_H
00023
00024 #include <volk_gnsssdr/volk_gnsssdr.h>
00025
00026 /** \addtogroup Tracking
00027  * \{ */
00028 /** \addtogroup Tracking_libs
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief Class that implements carrier wipe-off and correlators.
00034  */
00035 class Cpu_Multicorrelator_16sc
00036 {
00037 public:
00038     Cpu_Multicorrelator_16sc() = default;
00039     ~Cpu_Multicorrelator_16sc();
00040     bool init(int max_signal_length_samples, int n_correlators);
00041     bool set_local_code_and_taps(int code_length_chips, const lv_16sc_t *local_code_in, float
      *shifts_chips);
00042     bool set_input_output_vectors(lv_16sc_t *corr_out, const lv_16sc_t *sig_in);
00043     void update_local_code(int correlator_length_samples, float rem_code_phase_chips, float
      code_phase_step_chips);
00044     bool Carrier_wipeoff_multicorrelator_resampler(float rem_carrier_phase_in_rad, float
      phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, int signal_length_samples);
00045     bool free();
00046
00047 private:
00048     // Allocate the device input vectors
00049     const lv_16sc_t *d_sig_in{nullptr};
00050     const lv_16sc_t *d_local_code_in{nullptr};
00051     lv_16sc_t **d_local_codes_resampled{nullptr};
00052     lv_16sc_t *d_corr_out{nullptr};
00053     float *d_shifts_chips{nullptr};
00054     int d_code_length_chips{0};
00055     int d_n_correlators{0};
00056 };
00057
00058
00059 /** \} */
00060 /** \} */
00061 #endif  // GNSS_SDR_CPU_MULTICORRELATOR_H
```

13.512 cpu_multicorrelator_real_codes.h File Reference

Highly optimized CPU vector multiTAP correlator class using real-valued local codes.

`#include <complex>`

Classes

- class Cpu_Multicorrelator_Real_Codes

  Class that implements carrier wipe-off and correlators.

13.512.1 Detailed Description

Highly optimized CPU vector multiTAP correlator class using real-valued local codes.

Authors

- Javier Arribas, 2015. jarribas(at)cttc.es
- Cillian O'Driscoll, 2017. cillian.odriscoll(at)gmail.com
- Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

Class that implements a highly optimized vector multiTAP correlator class for CPUs
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cpu_multicorrelator_real_codes.h.

13.513 cpu_multicorrelator_real_codes.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cpu_multicorrelator_real_codes.h
00003  * \brief Highly optimized CPU vector multiTAP correlator class using real-valued local codes
00004  * \authors <ul>
00005  *          <li> Javier Arribas, 2015. jarribas(at)cttc.es
00006  *          <li> Cillian O'Driscoll, 2017. cillian.odriscoll(at)gmail.com
00007  *          <li> Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00008  *          </ul>
00009  *
00010  * Class that implements a highly optimized vector multiTAP correlator class for CPUs
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_CPU_MULTICORRELATOR_REAL_CODES_H
00024 #define GNSS_SDR_CPU_MULTICORRELATOR_REAL_CODES_H
00025
00026
00027 #include <complex>
00028
00029 /** \addtogroup Tracking
00030  * \{ */
00031 /** \addtogroup Tracking_libs
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief Class that implements carrier wipe-off and correlators.
00037  */
00038 class Cpu_Multicorrelator_Real_Codes
00039 {
00040 public:
00041     Cpu_Multicorrelator_Real_Codes() = default;
00042     void set_high_dynamics_resampler(bool use_high_dynamics_resampler);
00043     ~Cpu_Multicorrelator_Real_Codes();
00044     bool init(int max_signal_length_samples, int n_correlators);
00045     bool init(int max_signal_length_samples, int n_correlators, bool has_data_prompt);
00046     bool set_local_code_and_taps(int code_length_chips, const float *local_code_in, float
      *shifts_chips);
00047     bool set_input_output_vectors(std::complex<float> *corr_out, const std::complex<float> *sig_in);
00048     bool set_data_code_and_prompt_tap(int code_length_chips, const float *data_code_in, float
      *prompt_shift_chips);
00049     bool set_input_output_vectors(std::complex<float> *corr_out, std::complex<float> *data_prompt_out,
      const std::complex<float> *sig_in);
00050     void update_local_code(int correlator_length_samples, float rem_code_phase_chips, float
      code_phase_step_chips, float code_phase_rate_step_chips = 0.0);
00051     bool Carrier_wipeoff_multicorrelator_resampler(float rem_carrier_phase_in_rad, float
      phase_step_rad, float phase_rate_step_rad, float rem_code_phase_chips, float code_phase_step_chips,
      float code_phase_rate_step_chips, int signal_length_samples);
00052     bool Carrier_wipeoff_multicorrelator_resampler(float rem_carrier_phase_in_rad, float
      phase_step_rad, float rem_code_phase_chips, float code_phase_step_chips, float
      code_phase_rate_step_chips, int signal_length_samples);
```

```
00053     bool Carrier_wipeoff_multicorrelator_resampler_with_data_prompt(float rem_carrier_phase_in_rad,
      float phase_step_rad, float phase_rate_step_rad, float rem_code_phase_chips, float
      code_phase_step_chips, float code_phase_rate_step_chips, int signal_length_samples);
00054     bool free();
00055
00056 private:
00057     // Allocate the device input vectors
00058     const std::complex<float> *d_sig_in{nullptr};
00059     const float *d_local_code_in{nullptr};
00060     const float *d_data_code_in{nullptr};
00061     std::complex<float> *d_corr_out{nullptr};
00062     std::complex<float> *d_data_prompt_out{nullptr};
00063     std::complex<float> *d_corr_out_with_data_prompt{nullptr};
00064     float **d_local_codes_resampled{nullptr};
00065     float *d_shifts_chips{nullptr};
00066     float *d_data_prompt_shift_chips{nullptr};
00067     int d_code_length_chips{0};
00068     int d_data_code_length_chips{0};
00069     int d_n_correlators{0};
00070     int d_n_correlators_with_data_prompt{0};
00071     bool d_has_data_prompt{false};
00072     bool d_use_high_dynamics_resampler{true};
00073 };
00074
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_CPU_MULTICORRELATOR_REAL_CODES_H
```

13.514 cuda_multicorrelator.h File Reference

Highly optimized CUDA GPU vector multiTAP correlator class.

```
#include <complex>
#include <cuda.h>
#include <cuda_runtime.h>
```

Classes

- struct GPU_Complex
- struct GPU_Complex_Short
- class cuda_multicorrelator

  Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators.

13.514.1 Detailed Description

Highly optimized CUDA GPU vector multiTAP correlator class.

Authors

- Javier Arribas, 2015. jarribas(at)cttc.es

Class that implements a highly optimized vector multiTAP correlator class for NVIDIA CUDA GPUs
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file cuda_multicorrelator.h.

13.514.2 Macro Definition Documentation

13.514.2.1 CUDA_CALLABLE_MEMBER_DEVICE

```
#define CUDA_CALLABLE_MEMBER_DEVICE
```

Definition at line 30 of file cuda_multicorrelator.h.

13.514.2.2 CUDA_CALLABLE_MEMBER_GLOBAL

```
#define CUDA_CALLABLE_MEMBER_GLOBAL
```

Definition at line 29 of file cuda_multicorrelator.h.

13.515 cuda_multicorrelator.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file cuda_multicorrelator.h
00003  * \brief Highly optimized CUDA GPU vector multiTAP correlator class
00004  * \authors <ul>
00005  *          <li> Javier Arribas, 2015. jarribas(at)cttc.es
00006  *          </ul>
00007  *
00008  * Class that implements a highly optimized vector multiTAP correlator class for NVIDIA CUDA GPUs
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CUDA_MULTICORRELATOR_H
00022 #define GNSS_SDR_CUDA_MULTICORRELATOR_H
00023
00024
00025 #ifdef __CUDACC__
00026 #define CUDA_CALLABLE_MEMBER_GLOBAL __global__
00027 #define CUDA_CALLABLE_MEMBER_DEVICE __device__
00028 #else
00029 #define CUDA_CALLABLE_MEMBER_GLOBAL
00030 #define CUDA_CALLABLE_MEMBER_DEVICE
00031 #endif
00032
00033 #include <complex>
00034 #include <cuda.h>
00035 #include <cuda_runtime.h>
00036
00037 /** \addtogroup Tracking
00038  * \{ */
00039 /** \addtogroup Tracking_libs
00040  * \{ */
00041
00042
00043 // GPU new internal data types for complex numbers
00044
00045 struct GPU_Complex
00046 {
00047     float r;
00048     float i;
00049     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex() {};
00050     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex(float a, float b) : r(a), i(b) {}
00051     CUDA_CALLABLE_MEMBER_DEVICE float magnitude2(void) const { return r * r + i * i; }
00052     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex operator*(const GPU_Complex& a) const
00053     {
00054 #ifdef __CUDACC__
00055         return GPU_Complex(__fmul_rn(r, a.r) - __fmul_rn(i, a.i), __fmul_rn(i, a.r) + __fmul_rn(r,
      a.i));
00056 #else
00057         return GPU_Complex(r * a.r - i * a.i, i * a.r + r * a.i);
00058 #endif
00059     }
00060     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex operator+(const GPU_Complex& a) const
00061     {
00062         return GPU_Complex(r + a.r, i + a.i);
00063     }
00064     CUDA_CALLABLE_MEMBER_DEVICE void operator+=(const GPU_Complex& a)
00065     {
00066         r += a.r;
00067         i += a.i;
00068     }
00069     CUDA_CALLABLE_MEMBER_DEVICE void multiply_acc(const GPU_Complex& a, const GPU_Complex& b)
00070     {
00071         // c=a*b+c
00072         // real part
00073         // c.r=(a.r*b.r - a.i*b.i)+c.r
00074 #ifdef __CUDACC__
00075         r = __fmaf_rn(a.r, b.r, r);
00076         r = __fmaf_rn(-a.i, b.i, r);
00077         // imag part
00078         i = __fmaf_rn(a.i, b.r, i);
00079         i = __fmaf_rn(a.r, b.i, i);
00080 #else
00081         r = (a.r * b.r - a.i * b.i) + r;
00082         i = (a.i * b.r - a.r * b.i) + i;
```

```
00083 #endif
00084     }
00085 };
00086
00087
00088 struct GPU_Complex_Short
00089 {
00090     float r;
00091     float i;
00092     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short(short int a, short int b) : r(a), i(b) {}
00093     CUDA_CALLABLE_MEMBER_DEVICE float magnitude2(void) const
00094     {
00095         return r * r + i * i;
00096     }
00097     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short operator*(const GPU_Complex_Short& a) const
00098     {
00099         return GPU_Complex_Short(r * a.r - i * a.i, i * a.r + r * a.i);
00100     }
00101     CUDA_CALLABLE_MEMBER_DEVICE GPU_Complex_Short operator+(const GPU_Complex_Short& a) const
00102     {
00103         return GPU_Complex_Short(r + a.r, i + a.i);
00104     }
00105 };
00106
00107
00108 /*!
00109  * \brief Class that implements carrier wipe-off and correlators using NVIDIA CUDA GPU accelerators.
00110  */
00111 class cuda_multicorrelator
00112 {
00113 public:
00114     cuda_multicorrelator();
00115     bool init_cuda_integrated_resampler(
00116         int signal_length_samples,
00117         int code_length_chips,
00118         int n_correlators);
00119     bool set_local_code_and_taps(
00120         int code_length_chips,
00121         const std::complex<float>* local_codes_in,
00122         float* shifts_chips,
00123         int n_correlators);
00124     bool set_input_output_vectors(
00125         std::complex<float>* corr_out,
00126         std::complex<float>* sig_in);
00127
00128     bool free_cuda();
00129     bool Carrier_wipeoff_multicorrelator_resampler_cuda(
00130         float rem_carrier_phase_in_rad,
00131         float phase_step_rad,
00132         float code_phase_step_chips,
00133         float rem_code_phase_chips,
00134         int signal_length_samples,
00135         int n_correlators);
00136
00137 private:
00138     cudaStream_t stream1;
00139     // cudaStream_t stream2;
00140
00141     // Allocate the device input vectors
00142     GPU_Complex* d_sig_in;
00143     GPU_Complex* d_nco_in;
00144     GPU_Complex* d_sig_doppler_wiped;
00145     GPU_Complex* d_local_codes_in;
00146     GPU_Complex* d_corr_out;
00147
00148     std::complex<float>* d_sig_in_cpu;
00149     std::complex<float>* d_corr_out_cpu;
00150
00151     float* d_shifts_chips;
00152     int* d_shifts_samples;
00153     int d_code_length_chips;
00154
00155     int selected_gps_device;
00156     int threadsPerBlock;
00157     int blocksPerGrid;
00158
00159     int num_gpu_devices;
00160     int selected_device;
00161 };
00162
00163
00164 /** \} */
00165 /** \} */
00166 #endif  // GNSS_SDR_CUDA_MULTICORRELATOR_H
```

13.516 dll_pll_conf.h File Reference

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL.

```
#include "configuration_interface.h"
#include <cstdint>
#include <string>
```

Classes

- class Dll_Pll_Conf

13.516.1 Detailed Description

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL.

Author

Javier Arribas, 2018. jarribas(at)cttc.es

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dll_pll_conf.h.

13.517 dll_pll_conf.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_conf.h
00003  * \brief Class that contains all the configuration parameters for generic tracking block based on a
      DLL and a PLL.
00004  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00005  *
00006  * Class that contains all the configuration parameters for generic tracking block based on a DLL and
      a PLL.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_DLL_PLL_CONF_H
00020 #define GNSS_SDR_DLL_PLL_CONF_H
00021
00022 #include "configuration_interface.h"
00023 #include <cstdint>
00024 #include <string>
00025
00026 /** \addtogroup Tracking
00027  * \{ */
00028 /** \addtogroup Tracking_libs
00029  * \{ */
00030
00031
00032 class Dll_Pll_Conf
00033 {
00034 public:
00035     Dll_Pll_Conf();
00036     void SetFromConfiguration(const ConfigurationInterface *configuration, const std::string &role);
00037
00038     /* DLL/PLL tracking configuration */
00039     std::string item_type{"gr_complex"};
00040     std::string dump_filename{"./dll_pll_dump.dat"};
00041     std::string f_error_dump_filename{"./f_error_dump.csv"};
00042     double fs_in{2000000.0};
00043     double carrier_lock_th{0.0};
00044     double bs_dominance_ratio{0.5};
00045     double bs_runner_up_margin{0.3};
```

```
00046     double bs_transition_confidence{0.6};
00047     float pll_pull_in_bw_hz{50.0};
00048     float dll_pull_in_bw_hz{3.0};
00049     float fll_bw_hz{35.0};
00050     float pll_bw_hz{35.0};
00051     float dll_bw_hz{2.0};
00052     float pll_bw_narrow_hz{5.0};
00053     float dll_bw_narrow_hz{0.75};
00054     float early_late_space_chips{0.25};
00055     float very_early_late_space_chips{0.5};
00056     float early_late_space_narrow_chips{0.15};
00057     float very_early_late_space_narrow_chips{0.5};
00058     float slope{1.0};
00059     float spc{0.5};
00060     float y_intercept{1.0};
00061     float cn0_smoother_alpha{0.002};
00062     float carrier_lock_test_smoother_alpha{0.002};
00063     float bs_min_prompt_mag{0.0};
00064     uint32_t pull_in_time_s{5U};
00065     uint32_t bit_synchronization_time_limit_s{20U};
00066     uint32_t vector_length{0U};
00067     uint32_t smoother_length{10U};
00068     uint32_t f_error_accumulation{20U};
00069     uint32_t f_error_step_num{0U};
00070     double f_error_doppler_step{250.0};
00071     int32_t fll_filter_order{1};
00072     int32_t pll_filter_order{3};
00073     int32_t dll_filter_order{2};
00074     int32_t extend_correlation_symbols{1};
00075     int32_t cn0_samples{0};
00076     int32_t cn0_smoother_samples{200};
00077     int32_t carrier_lock_test_smoother_samples{25};
00078     int32_t cn0_min{0};
00079     int32_t max_code_lock_fail{0};
00080     int32_t max_carrier_lock_fail{0};
00081     int32_t bs_stable_best_required{3};
00082     int32_t bs_min_events_for_lock{6};
00083     int32_t bs_transition_window_epochs{4};
00084     int32_t bs_tentative_events_required{2};
00085     char signal[3]{};
00086     char system{'G'};
00087     bool enable_fll_pull_in{false};
00088     bool enable_fll_steady_state{false};
00089     bool track_pilot{true};
00090     bool enable_doppler_correction{false};
00091     bool carrier_aiding{true};
00092     bool high_dyn{false};
00093     bool dump{false};
00094     bool dump_mat{true};
00095     bool f_error_dump{false};
00096     bool tow_to_trk{false};
00097     bool bs_use_phase_dot_detector{true};
00098     // BeiDou B1C-specific options.
00099     bool qmboc{true};  //!< QMBOC local replica (same key as Acquisition_*.qmboc).
00100     bool b1c_prompt_use_data_q{true};
00101     bool b1c_prompt_normalize_power{true};
00102     float b1c_data_prompt_scale{1.7320508F};  // sqrt(3), compensates 1:3 data:pilot power.
00103     float b1c_pilot_prompt_scale{1.0F};
00104     float b1c_secondary_lock_ratio{0.88F};
00105 };
00106
00107
00108 /** \} */
00109 /** \} */
00110 #endif  // GNSS_SDR_DLL_PLL_CONF_H
```

13.518 dll_pll_conf_fpga.h File Reference

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL for the FPGA.

```
#include "configuration_interface.h"
#include <cstdint>
#include <string>
```

Classes

- class Dll_Pll_Conf_Fpga

13.518.1 Detailed Description

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL for the FPGA.

Author

Marc Majoral, 2019. mmajoral(at)cttc.cat

Javier Arribas, 2018. jarribas(at)cttc.es

Class that contains all the configuration parameters for generic tracking block based on a DLL and a PLL.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file dll_pll_conf_fpga.h.

13.519 dll_pll_conf_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file dll_pll_conf_fpga.h
00003  * \brief Class that contains all the configuration parameters for generic
00004  * tracking block based on a DLL and a PLL for the FPGA.
00005  * \author Marc Majoral, 2019. mmajoral(at)cttc.cat
00006  * \author Javier Arribas, 2018. jarribas(at)cttc.es
00007  *
00008  * Class that contains all the configuration parameters for generic tracking block based on a DLL and
     a PLL.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_DLL_PLL_CONF_FPGA_H
00022 #define GNSS_SDR_DLL_PLL_CONF_FPGA_H
00023
00024 #include "configuration_interface.h"
00025 #include <cstdint>
00026 #include <string>
00027
00028 /** \addtogroup Tracking
00029  * \{ */
00030 /** \addtogroup Tracking_libs
00031  * \{ */
00032
00033
00034 class Dll_Pll_Conf_Fpga
00035 {
00036 public:
00037     Dll_Pll_Conf_Fpga();
00038     void SetFromConfiguration(const ConfigurationInterface* configuration, const std::string& role);
00039
00040     /* DLL/PLL tracking configuration */
00041     std::string device_name{"/dev/uio"};
00042     std::string dump_filename{"./dll_pll_dump.dat"};
00043
00044     double fs_in{12500000.0};
00045     double carrier_lock_th{0.0};
00046     double bs_dominance_ratio{0.5};
00047     double bs_runner_up_margin{0.3};
00048     double bs_transition_confidence{0.6};
00049
00050     float pll_pull_in_bw_hz{50.0};
00051     float dll_pull_in_bw_hz{3.0};
00052     float fll_bw_hz{35.0};
00053     float pll_bw_hz{5.0};
00054     float dll_bw_hz{0.5};
00055     float pll_bw_narrow_hz{2.0};
00056     float dll_bw_narrow_hz{0.25};
00057     float early_late_space_chips{0.25};
00058     float very_early_late_space_chips{0.5};
00059     float early_late_space_narrow_chips{0.15};
00060     float very_early_late_space_narrow_chips{0.5};
00061     float slope{1.0};
```

```
00062     float spc{0.5};
00063     float y_intercept{1.0};
00064     float cn0_smoother_alpha{0.002};
00065     float carrier_lock_test_smoother_alpha{0.002};
00066     float bs_min_prompt_mag{0.0};
00067
00068     uint32_t pull_in_time_s{5U};  // signed integer, when pull in time is not yet reached it has to be
      compared against a negative number
00069     uint32_t bit_synchronization_time_limit_s{70U};
00070     uint32_t vector_length{0U};
00071     uint32_t smoother_length{10U};
00072     uint32_t code_length_chips{0U};
00073     uint32_t code_samples_per_chip{0U};
00074     uint32_t extend_fpga_integration_periods{1};
00075     uint32_t fpga_integration_period{0};
00076
00077     int32_t fll_filter_order{1};
00078     int32_t pll_filter_order{3};
00079     int32_t dll_filter_order{2};
00080     int32_t extend_correlation_symbols{1};
00081     int32_t cn0_samples{0};
00082     int32_t cn0_min{0};
00083     int32_t max_code_lock_fail{0};
00084     int32_t max_carrier_lock_fail{0};
00085     int32_t cn0_smoother_samples{200};
00086     int32_t carrier_lock_test_smoother_samples{25};
00087     // int32_t max_lock_fail;
00088     int32_t bs_stable_best_required{3};
00089     int32_t bs_min_events_for_lock{6};
00090     int32_t bs_transition_window_epochs{4};
00091     int32_t bs_tentative_events_required{2};
00092
00093     int32_t* ca_codes{nullptr};
00094     int32_t* data_codes{nullptr};
00095
00096     char signal[3]{};
00097     char system{'G'};
00098
00099     bool extended_correlation_in_fpga{false};
00100     bool track_pilot{true};
00101     bool enable_doppler_correction{false};
00102     bool enable_fll_pull_in{false};
00103     bool enable_fll_steady_state{false};
00104     bool carrier_aiding{true};
00105     bool high_dyn{false};
00106     bool dump{false};
00107     bool dump_mat{true};
00108     bool bs_use_phase_dot_detector{true};
00109 };
00110
00111
00112 /** \} */
00113 /** \} */
00114 #endif  // GNSS_SDR_DLL_PLL_CONF_FPGA_H
```

13.520 exponential_smoother.h File Reference

Class that implements an exponential smoother.

`#include <vector>`

Classes

- class Exponential_Smoother

  Class that implements a first-order exponential smoother.

13.520.1 Detailed Description

Class that implements an exponential smoother.

Authors

Carles Fernandez, 2019 cfernandez@cttc.es

Class that implements a first-order exponential smoother.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file exponential_smoother.h.

13.521 exponential_smoother.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file exponential_smoother.h
00003  * \brief Class that implements an exponential smoother
00004  * \authors Carles Fernandez, 2019 cfernandez@cttc.es
00005  *
00006  * Class that implements a first-order exponential smoother.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_EXPONENTIAL_SMOOTHER_H
00021 #define GNSS_SDR_EXPONENTIAL_SMOOTHER_H
00022
00023 #include <vector>
00024
00025 /** \addtogroup Tracking
00026  * \{ */
00027 /** \addtogroup Tracking_libs
00028  * \{ */
00029
00030
00031 /*! \brief
00032  * Class that implements a first-order exponential smoother.
00033  *
00034  * smoothed_value[k] = alpha * raw + (1-alpha) * smoothed_value[k-1]
00035  *
00036  * The length of the initialization can be controlled with
00037  * set_samples_for_initialization(int num_samples)
00038  */
00039 class Exponential_Smoother
00040 {
00041 public:
00042     Exponential_Smoother();             //!< Constructor
00043     ~Exponential_Smoother() = default;  //!< Destructor
00044
00045     Exponential_Smoother(Exponential_Smoother&&) = default;                      //!< Move operator
00046     Exponential_Smoother& operator=(Exponential_Smoother&& /*other*/) = default;  //!< Move assignment
      operator
00047
00048     void set_alpha(float alpha);                            //!< 0 < alpha < 1. The higher, the most
      responsive, but more variance. Default value: 0.001
00049     void set_samples_for_initialization(int num_samples);  //!< Number of samples averaged for
      initialization. Default value: 200
00050     void reset();
00051     void set_min_value(float value);
00052     void set_offset(float offset);
00053     float smooth(float raw);
00054     double smooth(double raw);
00055
00056 private:
00057     std::vector<float> init_buffer_;
00058     float alpha_{0.001};
00059     float one_minus_alpha_{0.999};
00060     float old_value_{0.0};
00061     float min_value_{25.0};
00062     float offset_{12.0};
00063     int samples_for_initialization_{200};
00064     int init_counter_{0};
00065     bool initializing_{true};
00066 };
00067
00068
00069 /** \} */
00070 /** \} */
00071 #endif  // GNSS_SDR_EXPONENTIAL_SMOOTHER_H
```

13.522 fpga_multicorrelator.h File Reference

FPGA vector correlator class.

```
#include <gnuradio/block.h>
#include <volk_gnsssdr/volk_gnsssdr_alloc.h>
#include <cstdint>
#include <string>
```

Classes

- class Fpga_Multicorrelator_8sc

  Class that implements carrier wipe-off and correlators.

13.522.1 Detailed Description

FPGA vector correlator class.

Authors

- Marc Majoral, 2019. mmajoral(at)cttc.cat
- Javier Arribas, 2019. jarribas(at)cttc.es

Class that controls and executes a highly optimized vector correlator class in the FPGA
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file fpga_multicorrelator.h.

13.523 fpga_multicorrelator.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file fpga_multicorrelator.h
00003  * \brief FPGA vector correlator class
00004  * \authors <ul>
00005  *          <li> Marc Majoral, 2019. mmajoral(at)cttc.cat
00006  *          <li> Javier Arribas, 2019. jarribas(at)cttc.es
00007  *          </ul>
00008  *
00009  * Class that controls and executes a highly optimized vector correlator
00010  * class in the FPGA
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_FPGA_MULTICORRELATOR_H
00024 #define GNSS_SDR_FPGA_MULTICORRELATOR_H
00025
00026 #include <gnuradio/block.h>
00027 #include <volk_gnsssdr/volk_gnsssdr_alloc.h>
00028 #include <cstdint>
00029 #include <string>
00030
00031 /** \addtogroup Tracking
00032  * \{ */
00033 /** \addtogroup Tracking_libs
00034  * \{ */
00035
00036
00037 /*!
00038  * \brief Class that implements carrier wipe-off and correlators.
00039  */
00040 class Fpga_Multicorrelator_8sc
00041 {
00042 public:
00043     /*!
00044      * \brief Constructor
00045      */
00046     Fpga_Multicorrelator_8sc(int32_t n_correlators,
```

```
00047         int32_t *ca_codes,
00048         int32_t *data_codes,
00049         uint32_t code_length_chips,
00050         bool track_pilot,
00051         uint32_t code_samples_per_chip);
00052
00053     /*!
00054      * \brief Destructor
00055      */
00056     ~Fpga_Multicorrelator_8sc();
00057
00058     /*!
00059      * \brief Configure pointers to the FPGA multicorrelator results
00060      */
00061     void set_output_vectors(gr_complex *corr_out, gr_complex *Prompt_Data);
00062
00063     /*!
00064      * \brief Configure the local code in the FPGA multicorrelator
00065      */
00066     void set_local_code_and_taps(
00067         float *shifts_chips, float *prompt_data_shift, int32_t PRN);
00068
00069     /*!
00070      * \brief Configure code phase and code rate parameters in the FPGA
00071      */
00072     void update_local_code();
00073
00074     /*!
00075      * \brief Perform a multicorrelation
00076      */
00077     void Carrier_wipeoff_multicorrelator_resampler(
00078         float rem_carrier_phase_in_rad,
00079         float phase_step_rad,
00080         float carrier_phase_rate_step_rad,
00081         float rem_code_phase_chips,
00082         float code_phase_step_chips,
00083         float code_phase_rate_step_chips,
00084         int32_t signal_length_samples);
00085
00086     /*!
00087      * \brief Stop the correlation process in the FPGA and free code phase and code rate parameters
00088      */
00089     bool free();
00090
00091     /*!
00092      * \brief Open the FPGA device driver
00093      */
00094     void open_channel(const std::string &device_io_name, uint32_t channel);
00095
00096     /*!
00097      * \brief Set the initial sample number where the tracking process begins
00098      */
00099     void set_initial_sample(uint64_t samples_offset);
00100
00101     /*!
00102      * \brief Read the sample counter in the FPGA
00103      */
00104     uint64_t read_sample_counter();
00105
00106     /*!
00107      * \brief Start the tracking process in the FPGA
00108      */
00109     void lock_channel();
00110
00111     /*!
00112      * \brief finish the tracking process in the FPGA
00113      */
00114     void unlock_channel();
00115
00116     /*!
00117      * \brief Set the secondary code length in the FPGA. This is only used when extended coherent
      integration
00118      * is enabled in the FPGA. If tracking the pilot is enabled then secondary_code_0_length is the
      length of the pilot
00119      * secondary code and secondary_code_1_length is the length of the data secondary code. If
      tracking the pilot is disabled
00120      * then secondary_code_0_length is the length of the data secondary code, and
      secondary_code_1_length must be set to zero.
00121      */
00122     void set_secondary_code_lengths(uint32_t secondary_code_0_length, uint32_t
      secondary_code_1_length);
00123
00124     /*!
00125      * \brief Initialize the secondary code in the FPGA. If tracking the pilot is enabled then the
      pilot secondary code is
00126      * configured when secondary_code = 0 and the data secondary code is configured when
      secondary_code = 1. If tracking the
```

```
00127      * pilot is disabled then the data secondary code is configured when secondary code = 0.
00128      */
00129     void initialize_secondary_code(uint32_t secondary_code, std::string *secondary_code_string);
00130
00131     /*!
00132      * \brief Set the PRN length in the FPGA in number of samples. This function is only used then
      extended coherent integration is enabled in the
00133      * FPGA. The FPGA allows for the configuration of two PRN lengths. When the length of the extended
      coherent integration is bigger than the
00134      * length of the PRN code, the FPGA uses the first_length_secondary_code as the length of the PRN
      code immediately following the beginning
00135      * of the extended coherent integration, and the next_length_secondary_code as the length of the
      remaining PRN codes.
00136      * The purpose of this is to have the option to allow the FPGA to compensate for a possible
      deviation between the nominal value of the PRN
00137      * code length and the measured PRN code length in the PRN immediately following the start of the
      coherent integration only.
00138      * If this option is not used then write the same value to first_length_secondary_code  and
      next_length_secondary_code.
00139      */
00140     void update_prn_code_length(uint32_t first_prn_length, uint32_t next_prn_length);
00141
00142     /*!
00143      * \brief Enable the use of secondary codes in the FPGA
00144      */
00145     void enable_secondary_codes();
00146
00147     /*!
00148      * \brief Disable the use of secondary codes in the FPGA
00149      */
00150     void disable_secondary_codes();
00151
00152 private:
00153     // FPGA register addresses
00154     // write addresses
00155     static const uint32_t code_phase_step_chips_num_reg_addr = 0;
00156     static const uint32_t initial_index_reg_base_addr = 1;
00157     static const uint32_t initial_interp_counter_reg_base_addr = 7;
00158     static const uint32_t nsamples_minus_1_reg_addr = 13;
00159     static const uint32_t code_length_minus_1_reg_addr = 14;
00160     static const uint32_t rem_carr_phase_rad_reg_addr = 15;
00161     static const uint32_t phase_step_rad_reg_addr = 16;
00162     static const uint32_t prog_mems_addr = 17;
00163     static const uint32_t drop_samples_reg_addr = 18;
00164     static const uint32_t initial_counter_value_reg_addr_lsw = 19;
00165     static const uint32_t initial_counter_value_reg_addr_msw = 20;
00166     static const uint32_t code_phase_step_chips_rate_reg_addr = 21;
00167     static const uint32_t phase_step_rate_reg_addr = 22;
00168     static const uint32_t stop_tracking_reg_addr = 23;
00169     static const uint32_t secondary_code_lengths_reg_addr = 25;
00170     static const uint32_t prog_secondary_code_0_data_reg_addr = 26;
00171     static const uint32_t prog_secondary_code_1_data_reg_addr = 27;
00172     static const uint32_t first_prn_length_minus_1_reg_addr = 28;
00173     static const uint32_t next_prn_length_minus_1_reg_addr = 29;
00174     static const uint32_t start_flag_addr = 30;
00175     // read-write addresses
00176     static const uint32_t test_reg_addr = 31;
00177     // read addresses
00178     static const uint32_t result_reg_real_base_addr = 1;
00179     static const uint32_t result_reg_imag_base_addr = 7;
00180     static const uint32_t sample_counter_reg_addr_lsw = 13;
00181     static const uint32_t sample_counter_reg_addr_msw = 14;
00182     // FPGA-related constants
00183     static const uint32_t secondary_code_word_size = 20;        // the secondary codes are written in
      to the FPGA in words of secondary_code_word_size bits
00184     static const uint32_t secondary_code_wr_strobe = 0x800000;  // write strobe position in the
      secondary code write register
00185     static const uint32_t secondary_code_addr_bits = 0x100000;  // memory address position in the
      secondary code write register
00186     static const uint32_t drop_samples = 1;                     // bit 0 of drop_samples_reg_addr
00187     static const uint32_t enable_secondary_code = 2;            // bit 1 of drop_samples_reg_addr
00188     static const uint32_t init_secondary_code_addresses = 4;    // bit 2 of drop_samples_reg_addr
00189     static const uint32_t FPGA_PAGE_SIZE = 0x1000;
00190     static const uint32_t max_code_resampler_counter = 1 « 31;  // 2^(number of bits of precision of
      the code resampler)
00191     static const uint32_t local_code_fpga_clear_address_counter = 0x10000000;
00192     static const uint32_t test_register_track_writeval = 0x55AA;
00193
00194     // private functions
00195     uint32_t fpga_acquisition_test_register(uint32_t writeval);
00196     void fpga_configure_tracking_gps_local_code(int32_t PRN);
00197     void fpga_compute_code_shift_parameters();
00198     void fpga_configure_code_parameters_in_fpga();
00199     void fpga_compute_signal_parameters_in_fpga();
00200     void fpga_configure_signal_parameters_in_fpga();
00201     void fpga_launch_multicorrelator_fpga();
00202     void read_tracking_gps_results();
```

```
00203     void close_device(void);
00204     void write_secondary_code(uint32_t secondary_code_length, std::string *secondary_code_string,
      uint32_t reg_addr);
00205
00206     volk_gnsssdr::vector<uint32_t> d_initial_index;
00207     volk_gnsssdr::vector<uint32_t> d_initial_interp_counter;
00208
00209     uint64_t d_initial_sample_counter;
00210
00211     gr_complex *d_corr_out;
00212     gr_complex *d_Prompt_Data;
00213
00214     float *d_shifts_chips;
00215     float *d_prompt_data_shift;
00216
00217     float d_rem_code_phase_chips;
00218     float d_code_phase_step_chips;
00219     float d_code_phase_rate_step_chips;
00220     float d_rem_carrier_phase_in_rad;
00221     float d_phase_step_rad;
00222     float d_carrier_phase_rate_step_rad;
00223
00224     uint32_t d_code_length_samples;
00225     uint32_t d_n_correlators;  // number of correlators
00226
00227     // data related to the hardware module and the driver
00228     int32_t d_device_descriptor;    // driver descriptor
00229     volatile uint32_t *d_map_base;  // driver memory map
00230
00231     // configuration data received from the interface
00232     uint32_t d_correlator_length_samples;
00233
00234     uint32_t d_code_phase_step_chips_num;
00235     uint32_t d_code_phase_rate_step_chips_num;
00236     int32_t d_rem_carr_phase_rad_int;
00237     int32_t d_phase_step_rad_int;
00238     int32_t d_carrier_phase_rate_step_rad_int;
00239
00240     // PRN codes
00241     int32_t *d_ca_codes;
00242     int32_t *d_data_codes;
00243
00244     // secondary code configuration
00245     uint32_t d_secondary_code_0_length;
00246     uint32_t d_secondary_code_1_length;
00247
00248     bool d_track_pilot;
00249     bool d_secondary_code_enabled;
00250 };
00251
00252
00253 /** \} */
00254 /** \} */
00255 #endif  // GNSS_SDR_FPGA_MULTICORRELATOR_H
```

13.524 kf_conf.h

```
00001 /*!
00002  * \file Kf_conf.h
00003  * \brief Class that contains all the configuration parameters for generic tracking block based on a
      Kalman Filter.
00004  * \author Javier Arribas, 2020. jarribas(at)cttc.es
00005  *
00006  * Class that contains all the configuration parameters for generic tracking block based on a DLL and
      a PLL.
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00011  *
00012  * GNSS-SDR is a software defined Global Navigation
00013  *          Satellite Systems receiver
00014  *
00015  * This file is part of GNSS-SDR.
00016  *
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022 #ifndef GNSS_SDR_Kf_CONF_H
00023 #define GNSS_SDR_Kf_CONF_H
00024
00025 #include "configuration_interface.h"
```

```
00026 #include <cstdint>
00027 #include <string>
00028
00029 class Kf_Conf
00030 {
00031 public:
00032     Kf_Conf();
00033     void SetFromConfiguration(const ConfigurationInterface *configuration, const std::string &role);
00034
00035     std::string item_type;
00036     std::string dump_filename;
00037     double fs_in;
00038     double carrier_lock_th;
00039
00040     // KF statistics
00041     // Measurement covariances (R)
00042     double code_disc_sd_chips;
00043     double carrier_disc_sd_rads;
00044
00045     // System covariances (Q)
00046     double code_phase_sd_chips;
00047     double carrier_phase_sd_rad;
00048     double carrier_freq_sd_hz;
00049     double carrier_freq_rate_sd_hz_s;
00050
00051     // initial Kalman covariance matrix (P)
00052     double init_code_phase_sd_chips;
00053     double init_carrier_phase_sd_rad;
00054     double init_carrier_freq_sd_hz;
00055     double init_carrier_freq_rate_sd_hz_s;
00056
00057     float early_late_space_chips;
00058     float very_early_late_space_chips;
00059     float early_late_space_narrow_chips;
00060     float very_early_late_space_narrow_chips;
00061     float slope;
00062     float spc;
00063     float y_intercept;
00064     float cn0_smoother_alpha;
00065     float carrier_lock_test_smoother_alpha;
00066     uint32_t pull_in_time_s;
00067     uint32_t bit_synchronization_time_limit_s;
00068     uint32_t vector_length;
00069     uint32_t smoother_length;
00070     int32_t extend_correlation_symbols;
00071     int32_t cn0_samples;
00072     int32_t cn0_smoother_samples;
00073     int32_t carrier_lock_test_smoother_samples;
00074     int32_t cn0_min;
00075     int32_t max_code_lock_fail;
00076     int32_t max_carrier_lock_fail;
00077     char signal[3]{};
00078     char system;
00079     bool track_pilot;
00080     bool enable_doppler_correction;
00081     bool high_dyn;
00082     bool dump;
00083     bool dump_mat;
00084 };
00085
00086 #endif
```

13.525 lock_detectors.h File Reference

Interface of a library with a set of code and carrier phase lock detectors.

`#include <gnuradio/gr_complex.h>`

Functions

- float cn0_svn_estimator (const gr_complex *Prompt_buffer, int length, float coh_integration_time_s)

  cn0_svn_estimator is a Carrier-to-Noise (CN0) estimator based on the Signal-to-Noise Variance (SNV) estimator
- float cn0_m2m4_estimator (const gr_complex *Prompt_buffer, int length, float coh_integration_time_s)

  cn0_m2m4_estimator is a Carrier-to-Noise (CN0) estimator based on the Second- and Fourth-Order Moments Method (M2M4)
- float carrier_lock_detector (const gr_complex *Prompt_buffer, int length)

  A carrier lock detector.

13.525.1 Detailed Description

Interface of a library with a set of code and carrier phase lock detectors.
SNV_CN0 is a Carrier-to-Noise (CN0) estimator based on the Signal-to-Noise Variance (SNV) estimator [1]. Carrier lock detector using normalised estimate of the cosine of twice the carrier phase error [2].
[1] Marco Pini, Emanuela Falletti and Maurizio Fantino, "Performance Evaluation of C/N0 Estimators using a Real Time GNSS Software Receiver," IEEE 10th International Symposium on Spread Spectrum Techniques and Applications, pp.28-30, August 2008.
[2] Van Dierendonck, A.J. (1996), Global Positioning System: Theory and Applications, Volume I, Chapter 8: GPS Receivers, AJ Systems, Los Altos, CA 94024. Inc.: 329-407.

Authors

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file lock_detectors.h.

13.526 lock_detectors.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file lock_detectors.h
00003  * \brief Interface of a library with a set of code and carrier phase lock detectors.
00004  *
00005  * SNV_CN0 is a Carrier-to-Noise (CN0) estimator
00006  * based on the Signal-to-Noise Variance (SNV) estimator [1].
00007  * Carrier lock detector using normalised estimate of the cosine
00008  * of twice the carrier phase error [2].
00009  *
00010  * [1] Marco Pini, Emanuela Falletti and Maurizio Fantino, "Performance
00011  * Evaluation of C/N0 Estimators using a Real Time GNSS Software Receiver,"
00012  * IEEE 10th International Symposium on Spread Spectrum Techniques and
00013  * Applications, pp.28-30, August 2008.
00014  *
00015  * [2] Van Dierendonck, A.J. (1996), Global Positioning System: Theory and
00016  * Applications,
00017  * Volume I, Chapter 8: GPS Receivers, AJ Systems, Los Altos, CA 94024.
00018  * Inc.: 329-407.
00019  * \authors <ul>
00020  *          <li> Javier Arribas, 2011. jarribas(at)cttc.es
00021  *          <li> Luis Esteve, 2012. luis(at)epsilon-formacion.com
00022  *          </ul>
00023  *
00024  * -----------------------------------------------------------------------------
00025  *
00026  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00027  * This file is part of GNSS-SDR.
00028  *
00029  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00030  * SPDX-License-Identifier: GPL-3.0-or-later
00031  *
00032  * -----------------------------------------------------------------------------
00033  */
00034
00035 #ifndef GNSS_SDR_LOCK_DETECTORS_H
00036 #define GNSS_SDR_LOCK_DETECTORS_H
00037
00038 #include <gnuradio/gr_complex.h>
00039
00040 /** \addtogroup Tracking
00041  * \{ */
00042 /** \addtogroup Tracking_libs
00043  * \{ */
00044
00045
00046 /*! \brief cn0_svn_estimator is a Carrier-to-Noise (CN0) estimator
00047  * based on the Signal-to-Noise Variance (SNV) estimator
00048  *
00049  * Signal-to-Noise (SNR) (\f$ \rho \f$) estimator using the Signal-to-Noise Variance (SNV) estimator:
00050  * \f{equation}
00051  *  \hat{\rho}=\frac{\hat{P}_s}{\hat{P}_n}=\frac{\hat{P}_s}{\hat{P}_{tot}-\hat{P}_s},
00052  * \f}
00053  *  where \f$ \hat{P}_s=\left(\frac{1}{N}\sum^{N-1}_{i=0}|Re(Pc(i))|\right)^2 \f$ is the estimation of
      the signal power,
```

```
00054  * \f$ \hat{P}_{tot}=\frac{1}{N}\sum^{N-1}_{i=0}|Pc(i)|^2 \f$ is the estimator of the total power, \f$
       |\cdot| \f$ is the absolute value,
00055  * \f$ Re(\cdot) \f$ stands for the real part of the value, and \f$ Pc(i) \f$ is the prompt correlator
       output for the sample index i.
00056  *
00057  * The SNR value is converted to CN0 [dB-Hz], taking into account the coherent integration time, using
       the following formula:
00058  * \f{equation}
00059  *     CN0_{dB}=10*log(\hat{\rho})-10*log(T_{int}),
00060  * \f}
00061  * where \f$ T_{int} \f$ is the coherent integration time, in seconds.
00062  *
00063  * Ref: Marco Pini, Emanuela Falletti and Maurizio Fantino, "Performance
00064  * Evaluation of C/N0 Estimators using a Real Time GNSS Software Receiver,"
00065  * IEEE 10th International Symposium on Spread Spectrum Techniques and
00066  * Applications, pp.28-30, August 2008.
00067  */
00068 float cn0_svn_estimator(const gr_complex* Prompt_buffer, int length, float coh_integration_time_s);
00069
00070
00071 /*! \brief cn0_m2m4_estimator is a Carrier-to-Noise (CN0) estimator
00072  * based on the Second- and Fourth-Order Moments Method (M2M4)
00073  *
00074  * Signal-to-Noise (SNR) (\f$ \rho \f$) estimator using the Moments Method:
00075  * \f{equation}
00076  *  \hat{\rho}=\frac{\sqrt{2 \hat{M}_2^2 - \hat{M}_4 }}{\hat{M}_2-\sqrt{2 \hat{M}_2^2 - \hat{M}_4 }},
00077  * \f}
00078  * where
00079  * \f$ \hat{M}_2=\frac{1}{N}\sum^{K-1}_{k=0}|P[k]|^2 \f$, \f$ \hat{M}_4 =
       \frac{1}{K}\sum^{K-1}_{k=0}|P[k]|^4 \f$, \f$ |\cdot| \f$ is the absolute value,
00080  * and \f$ P[k] \f$ is the prompt correlator output for the sample index k.
00081  *
00082  * The SNR value is converted to CN0 [dB-Hz] taking into account the coherent integration time, using
       the following formula:
00083  * \f{equation}
00084  *     CN0_{dB}=10*log(\hat{\rho})-10*log(T_{int}),
00085  * \f}
00086  * where \f$ T_{int} \f$ is the coherent integration time, in seconds.
00087  *
00088  * Ref: D. R. Pauluzzi, N. C. Beaulieu, "A comparison of SNR estimation
00089  * techniques for the AWGN channel," IEEE Trans. on Comm., vol. 48,
00090  * no. 10, pp. 1681-1691, Oct. 2000.
00091  */
00092 float cn0_m2m4_estimator(const gr_complex* Prompt_buffer, int length, float coh_integration_time_s);
00093
00094
00095 /*! \brief A carrier lock detector
00096  *
00097  * The Carrier Phase Lock Detector block uses the estimate of the cosine of twice the carrier phase
       error is given by
00098  * \f{equation}
00099  *     C2\phi=\frac{NBD}{NBP},
00100  * \f}
00101  *  where \f$ NBD=(\sum^{N-1}_{i=0}|Im(Pc(i))|)^2+(\sum^{N-1}_{i=0}|Re(Pc(i))|)^2 \f$,
00102  *  \f$ NBP=\sum^{N-1}_{i=0}Im(Pc(i))^2-\sum^{N-1}_{i=0}Re(Pc(i))^2 \f$, and
00103  *  \f$ Pc(i) \f$ is the prompt correlator output for the sample index i.
00104  * Ref: Van Dierendonck, A.J. (1996), Global Positioning System: Theory and
00105  * Applications,
00106  * Volume I, Chapter 8: GPS Receivers, AJ Systems, Los Altos, CA 94024.
00107  * Inc.: 329-407.
00108  */
00109 float carrier_lock_detector(const gr_complex* Prompt_buffer, int length);
00110
00111
00112 /** \} */
00113 /** \} */
00114 #endif  // GNSS_SDR_LOCK_DETECTORS_H
```

13.527 nonlinear_tracking.h File Reference

Interface of a library for nonlinear tracking algorithms.

```
#include <armadillo>
#include <gnuradio/gr_complex.h>
```

Classes

- class ModelFunction
- class CubatureFilter
- class UnscentedFilter

13.527.1 Detailed Description

Interface of a library for nonlinear tracking algorithms.
CubatureFilter implements the functionality of the Cubature Kalman Filter, which uses multidimensional cubature rules to estimate the time evolution of a nonlinear system. UnscentedFilter implements an Unscented Kalman Filter which uses Unscented Transform rules to perform a similar estimation.
[1] I Arasaratnam and S Haykin. Cubature kalman filters. IEEE Transactions on Automatic Control, 54(6):1254–1269,2009.

Authors

- Gerald LaMountain, 2019. gerald(at)ece.neu.edu
- Jordi Vila-Valls 2019. jvila(at)cttc.es

13.527.1.1

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file nonlinear_tracking.h.

13.528 nonlinear_tracking.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nonlinear_tracking.h
00003  * \brief Interface of a library for nonlinear tracking algorithms
00004  *
00005  * CubatureFilter implements the functionality of the Cubature Kalman
00006  * Filter, which uses multidimensional cubature rules to estimate the
00007  * time evolution of a nonlinear system. UnscentedFilter implements
00008  * an Unscented Kalman Filter which uses Unscented Transform rules to
00009  * perform a similar estimation.
00010  *
00011  * [1] I Arasaratnam and S Haykin. Cubature kalman filters. IEEE
00012  * Transactions on Automatic Control, 54(6):1254-1269,2009.
00013  *
00014  * \authors <ul>
00015  *          <li> Gerald LaMountain, 2019. gerald(at)ece.neu.edu
00016  *          <li> Jordi Vila-Valls 2019. jvila(at)cttc.es
00017  *          </ul>
00018  * -----------------------------------------------------------------------------
00019  *
00020  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00021  * This file is part of GNSS-SDR.
00022  *
00023  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00024  * SPDX-License-Identifier: GPL-3.0-or-later
00025  *
00026  * -----------------------------------------------------------------------------
00027  */
00028
00029 #ifndef GNSS_SDR_NONLINEAR_TRACKING_H
00030 #define GNSS_SDR_NONLINEAR_TRACKING_H
00031
00032 #if ARMA_NO_BOUND_CHECKING
00033 #define ARMA_NO_DEBUG 1
00034 #endif
00035
00036 #include <armadillo>
00037 #include <gnuradio/gr_complex.h>
00038
00039 /** \addtogroup Tracking
00040  * \{ */
00041 /** \addtogroup Tracking_libs
00042  * \{ */
00043
00044
00045 // Abstract model function
00046 class ModelFunction
00047 {
00048 public:
00049     ModelFunction() {};
00050     virtual arma::vec operator()(const arma::vec& input) = 0;
00051     virtual ~ModelFunction() = default;
00052 };
00053
00054 class CubatureFilter
```

```
00055 {
00056 public:
00057     // Constructors and destructors
00058     CubatureFilter();
00059     explicit CubatureFilter(int nx);
00060     CubatureFilter(const arma::vec& x_pred_0, const arma::mat& P_x_pred_0);
00061     ~CubatureFilter() = default;
00062
00063     // Reinitialization function
00064     void initialize(const arma::mat& x_pred_0, const arma::mat& P_x_pred_0);
00065
00066     // Prediction and estimation
00067     void predict_sequential(const arma::vec& x_post, const arma::mat& P_x_post, ModelFunction*
      transition_fcn, const arma::mat& noise_covariance);
00068     void update_sequential(const arma::vec& z_upd, const arma::vec& x_pred, const arma::mat& P_x_pred,
      ModelFunction* measurement_fcn, const arma::mat& noise_covariance);
00069
00070     // Getters
00071     arma::mat get_x_pred() const;
00072     arma::mat get_P_x_pred() const;
00073     arma::mat get_x_est() const;
00074     arma::mat get_P_x_est() const;
00075
00076 private:
00077     arma::vec x_pred_out;
00078     arma::mat P_x_pred_out;
00079     arma::vec x_est;
00080     arma::mat P_x_est;
00081 };
00082
00083 class UnscentedFilter
00084 {
00085 public:
00086     // Constructors and destructors
00087     UnscentedFilter();
00088     explicit UnscentedFilter(int nx);
00089     UnscentedFilter(const arma::vec& x_pred_0, const arma::mat& P_x_pred_0);
00090     ~UnscentedFilter() = default;
00091
00092     // Reinitialization function
00093     void initialize(const arma::mat& x_pred_0, const arma::mat& P_x_pred_0);
00094
00095     // Prediction and estimation
00096     void predict_sequential(const arma::vec& x_post, const arma::mat& P_x_post, ModelFunction*
      transition_fcn, const arma::mat& noise_covariance);
00097     void update_sequential(const arma::vec& z_upd, const arma::vec& x_pred, const arma::mat& P_x_pred,
      ModelFunction* measurement_fcn, const arma::mat& noise_covariance);
00098
00099     // Getters
00100     arma::mat get_x_pred() const;
00101     arma::mat get_P_x_pred() const;
00102     arma::mat get_x_est() const;
00103     arma::mat get_P_x_est() const;
00104
00105 private:
00106     arma::vec x_pred_out;
00107     arma::mat P_x_pred_out;
00108     arma::vec x_est;
00109     arma::mat P_x_est;
00110 };
00111
00112
00113 /** \} */
00114 /** \} */
00115 #endif  // GNSS_SDR_NONLINEAR_TRACKING_H
```

13.529 tcp_communication.h File Reference

Interface of the TCP communication class.

```
#include "tcp_packet_data.h"
#include <boost/array.hpp>
#include <boost/asio.hpp>
```

Classes

- class Tcp_Communication

  TCP communication class.

Macros

- #define NUM_TX_VARIABLES_GALILEO_E1 13
- #define NUM_TX_VARIABLES_GPS_L1_CA 9
- #define NUM_RX_VARIABLES 4

Typedefs

- using b_io_context = boost::asio::io_service

13.529.1 Detailed Description

Interface of the TCP communication class.

Author

David Pubill, 2011. dpubill(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file tcp_communication.h.

13.530 tcp_communication.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tcp_communication.h
00003  * \brief Interface of the TCP communication class
00004  * \author David Pubill, 2011. dpubill(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_TCP_COMMUNICATION_H
00019 #define GNSS_SDR_TCP_COMMUNICATION_H
00020
00021 #include "tcp_packet_data.h"
00022 #include <boost/array.hpp>
00023 #include <boost/asio.hpp>
00024
00025 /** \addtogroup Tracking
00026  * \{ */
00027 /** \addtogroup Tracking_libs
00028  * \{ */
00029
00030
00031 #if USE_BOOST_ASIO_IO_CONTEXT
00032 using b_io_context = boost::asio::io_context;
00033 #else
00034 using b_io_context = boost::asio::io_service;
00035 #endif
00036
00037 #define NUM_TX_VARIABLES_GALILEO_E1 13
00038 #define NUM_TX_VARIABLES_GPS_L1_CA 9
00039 #define NUM_RX_VARIABLES 4
00040
00041 /*!
00042  * \brief TCP communication class
00043  */
00044 class Tcp_Communication
00045 {
00046 public:
00047     Tcp_Communication();
00048     ~Tcp_Communication() = default;
00049
00050     int listen_tcp_connection(size_t d_port_, size_t d_port_ch0_);
00051     void send_receive_tcp_packet_galileo_e1(boost::array<float, NUM_TX_VARIABLES_GALILEO_E1> buf,
      Tcp_Packet_Data *tcp_data_);
```

```
00052     void send_receive_tcp_packet_gps_l1_ca(boost::array<float, NUM_TX_VARIABLES_GPS_L1_CA> buf,
      Tcp_Packet_Data *tcp_data_);
00053     void close_tcp_connection(size_t d_port_);
00054
00055 private:
00056     b_io_context io_context_;
00057     boost::asio::ip::tcp::socket tcp_socket_;
00058 };
00059
00060
00061 /** \} */
00062 /** \} */
00063 #endif
```

13.531 tcp_packet_data.h File Reference

Interface of the TCP data packet class.

Classes

- class Tcp_Packet_Data

  Class that implements a TCP data packet.

13.531.1 Detailed Description

Interface of the TCP data packet class.

Author

David Pubill, 2011. dpubill(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tcp_packet_data.h.

13.532 tcp_packet_data.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tcp_packet_data.h
00003  * \brief Interface of the TCP data packet class
00004  * \author David Pubill, 2011. dpubill(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_TCP_PACKET_DATA_H
00019 #define GNSS_SDR_TCP_PACKET_DATA_H
00020
00021 /** \addtogroup Tracking
00022  * \{ */
00023 /** \addtogroup Tracking_libs
00024  * \{ */
00025
00026
00027 /*!
00028  * \brief Class that implements a TCP data packet
00029  */
00030 class Tcp_Packet_Data
00031 {
00032 public:
00033     Tcp_Packet_Data() = default;
00034     ~Tcp_Packet_Data() = default;
00035     float proc_pack_code_error = 0.0;
00036     float proc_pack_carr_error = 0.0;
```

```
00037     float proc_pack_carrier_doppler_hz = 0.0;
00038 };
00039
00040
00041 /** \} */
00042 /** \} */
00043 #endif
```

13.533 tracking_2nd_DLL_filter.h File Reference

Interface of a 2nd order DLL filter for code tracking loop.

Classes

- class Tracking_2nd_DLL_filter

  This class implements a 2nd order DLL filter for code tracking loop.

13.533.1 Detailed Description

Interface of a 2nd order DLL filter for code tracking loop.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Class that implements a 2nd order PLL filter for code tracking loop. The algorithm is described in: K.Borre, D.M.↩ Akos, N.Bertelsen, P.Rinder, and S. H. Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007, Applied and Numerical Harmonic Analysis.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tracking_2nd_DLL_filter.h.

13.534 tracking_2nd_DLL_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_2nd_DLL_filter.h
00003  * \brief Interface of a 2nd order DLL filter for code tracking loop.
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * Class that implements a 2nd order PLL filter for code tracking loop.
00007  * The algorithm is described in:
00008  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S. H. Jensen,
00009  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach,
00010  * Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_TRACKING_2ND_DLL_FILTER_H
00024 #define GNSS_SDR_TRACKING_2ND_DLL_FILTER_H
00025
00026 /** \addtogroup Tracking
00027  * \{ */
00028 /** \addtogroup Tracking_libs
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class implements a 2nd order DLL filter for code tracking loop.
00034  *
00035  * The algorithm is described in:
00036  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S. H. Jensen, A Software-Defined GPS
00037  * and Galileo Receiver. A Single-Frequency Approach,
00038  * Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
```

```
00039  */
00040 class Tracking_2nd_DLL_filter
00041 {
00042 public:
00043     Tracking_2nd_DLL_filter();
00044     ~Tracking_2nd_DLL_filter() = default;
00045     explicit Tracking_2nd_DLL_filter(float pdi_code);
00046
00047     void set_DLL_BW(float dll_bw_hz);             //!< Set DLL filter bandwidth [Hz]
00048     void set_pdi(float pdi_code);                 //!< Set Summation interval for code [s]
00049     void initialize();                            //!< Start tracking with acquisition information
00050     float get_code_nco(float DLL_discriminator);  //!< Numerically controlled oscillator
00051
00052 private:
00053     void calculate_lopp_coef(float* tau1, float* tau2, float lbw, float zeta, float k);
00054
00055     // PLL filter parameters
00056     float d_tau1_code = 0.0;
00057     float d_tau2_code = 0.0;
00058     float d_pdi_code = 0.0;
00059     float d_dllnoisebandwidth = 0.0;
00060     float d_dlldampingratio = 0.0;
00061     float d_old_code_error = 0.0;
00062     float d_old_code_nco = 0.0;
00063 };
00064
00065
00066 /** \} */
00067 /** \} */
00068 #endif
```

13.535 tracking_2nd_PLL_filter.h File Reference

Interface of a 2nd order PLL filter for carrier tracking loop.

Classes

- class Tracking_2nd_PLL_filter

  This class implements a 2nd order PLL filter for carrier tracking loop.

13.535.1 Detailed Description

Interface of a 2nd order PLL filter for carrier tracking loop.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Class that implements 2 order PLL filter for tracking carrier loop. The algorithm is described in K.Borre, D.M.↩ Akos, N.Bertelsen, P.Rinder, and S.H. Jensen, A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach, Birkhauser, 2007, Applied and Numerical Harmonic Analysis.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tracking_2nd_PLL_filter.h.

13.536 tracking_2nd_PLL_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_2nd_PLL_filter.h
00003  * \brief Interface of a 2nd order PLL filter for carrier tracking loop
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * Class that implements 2 order PLL filter for tracking carrier loop.
00007  * The algorithm is described in
00008  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S.H. Jensen,
00009  * A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach,
00010  * Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
```

```
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_TRACKING_2ND_PLL_FILTER_H
00024 #define GNSS_SDR_TRACKING_2ND_PLL_FILTER_H
00025
00026 /** \addtogroup Tracking
00027  * \{ */
00028 /** \addtogroup Tracking_libs
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class implements a 2nd order PLL filter for carrier tracking loop.
00034  *
00035  * The algorithm is described in:
00036  * K.Borre, D.M.Akos, N.Bertelsen, P.Rinder, and S. H. Jensen, A Software-Defined GPS and Galileo
     Receiver. A Single-Frequency Approach,
00037  * Birkhauser, 2007, Applied and Numerical Harmonic Analysis.
00038  */
00039 class Tracking_2nd_PLL_filter
00040 {
00041 public:
00042     Tracking_2nd_PLL_filter();
00043     ~Tracking_2nd_PLL_filter() = default;
00044     explicit Tracking_2nd_PLL_filter(float pdi_carr);
00045
00046     void set_PLL_BW(float pll_bw_hz);  //!< Set PLL loop bandwidth [Hz]
00047     void set_pdi(float pdi_carr);      //!< Set Summation interval for code [s]
00048     void initialize();
00049     float get_carrier_nco(float PLL_discriminator);
00050
00051 private:
00052     void calculate_lopp_coef(float* tau1, float* tau2, float lbw, float zeta, float k);
00053     // PLL filter parameters
00054     float d_tau1_carr = 0.0;
00055     float d_tau2_carr = 0.0;
00056     float d_pdi_carr = 0.0;
00057     float d_pllnoisebandwidth = 0.0;
00058     float d_plldampingratio = 0.0;
00059     float d_old_carr_error = 0.0;
00060     float d_old_carr_nco = 0.0;
00061 };
00062
00063
00064 /** \} */
00065 /** \} */
00066 #endif
```

13.537 tracking_discriminators.h File Reference

Interface of a library with a set of code tracking and carrier tracking discriminators.

```
#include <gnuradio/gr_complex.h>
#include <cmath>
```

Functions

- double fll_four_quadrant_atan (gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2)
- double **fll_diff_atan** (gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2)
- double phase_unwrap (double phase_rad)

  Phase unwrapping function, input is [rad].
- double pll_four_quadrant_atan (gr_complex prompt_s1)

  PLL four quadrant arctan discriminator.
- double pll_cloop_two_quadrant_atan (gr_complex prompt_s1)

  PLL Costas loop two quadrant arctan discriminator.
- double dll_nc_e_minus_l_normalized (gr_complex early_s1, gr_complex late_s1, float spc=0.5, float slope=1.0, float y_intercept=1.0)

  DLL Noncoherent Early minus Late envelope normalized discriminator.

- double dll_nc_vemlp_normalized (gr_complex very_early_s1, gr_complex early_s1, gr_complex late_s1, gr_complex very_late_s1)

  DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator.
- template<typename Fun>
  double CalculateSlope (Fun &&f, double x)
- template<typename Fun>
  double CalculateSlopeAbs (Fun &&f, double x)
- template<typename Fun>
  double GetYIntercept (Fun &&f, double x)
- template<typename Fun>
  double GetYInterceptAbs (Fun &&f, double x)
- template<int M = 1, int N = M>
  double SinBocCorrelationFunction (double offset_in_chips)
- template<int M = 1, int N = M>
  double CosBocCorrelationFunction (double offset_in_chips)

13.537.1 Detailed Description

Interface of a library with a set of code tracking and carrier tracking discriminators.

Authors

- Javier Arribas, 2011. jarribas(at)cttc.es
- Luis Esteve, 2012. luis(at)epsilon-formacion.com

Library with a set of code tracking and carrier tracking discriminators that is used by the tracking algorithms.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file tracking_discriminators.h.

13.538 tracking_discriminators.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_discriminators.h
00003  * \brief Interface of a library with a set of code tracking and carrier
00004  * tracking discriminators.
00005  * \authors <ul>
00006  *          <li> Javier Arribas, 2011. jarribas(at)cttc.es
00007  *          <li> Luis Esteve, 2012. luis(at)epsilon-formacion.com
00008  *          </ul>
00009  *
00010  * Library with a set of code tracking and carrier tracking discriminators
00011  * that is used by the tracking algorithms.
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_TRACKING_DISCRIMINATORS_H
00025 #define GNSS_SDR_TRACKING_DISCRIMINATORS_H
00026
00027 #include <gnuradio/gr_complex.h>
00028 #include <cmath>
00029
00030 /** \addtogroup Tracking
00031  * \{ */
00032 /** \addtogroup Tracking_libs
00033  * \{ */
00034
00035
00036 /*! brief FLL four quadrant arctan discriminator
00037  *
00038  * FLL four quadrant arctan discriminator:
```

```
00039  * \f{equation}
00040  *     \frac{\phi_2-\phi_1}{t_2-t1}=\frac{ATAN2(cross,dot)}{t_1-t_2},
00041  * \f}
00042  * where \f$cross=I_{PS1}Q_{PS2}-I_{PS2}Q_{PS1}\f$ and \f$dot=I_{PS1}I_{PS2}+Q_{PS1}Q_{PS2}\f$,
00043  * \f$I_{PS1},Q_{PS1}\f$ are the inphase and quadrature prompt correlator outputs respectively at
       sample time \f$t_1\f$, and
00044  * \f$I_{PS2},Q_{PS2}\f$ are the inphase and quadrature prompt correlator outputs respectively at
       sample time \f$t_2\f$. The output is in [radians/second].
00045  */
00046 double fll_four_quadrant_atan(gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2);
00047
00048
00049 /*
00050  * FLL differential arctan discriminator:
00051  * \f{equation}
00052  *
       e_{atan}(k)=\frac{1}{t_1-t_2}\text{phase_unwrap}(\tan^-1(\frac{Q(k)}{I(k)})-\tan^-1(\frac{Q(k-1)}{I(k-1)}))
00053  * \f}
00054  * The output is in [radians/second].
00055  */
00056 double fll_diff_atan(gr_complex prompt_s1, gr_complex prompt_s2, double t1, double t2);
00057
00058
00059 /*! \brief Phase unwrapping function, input is [rad]
00060  */
00061 double phase_unwrap(double phase_rad);
00062
00063
00064 /*! \brief PLL four quadrant arctan discriminator
00065  *
00066  * PLL four quadrant arctan discriminator:
00067  * \f{equation}
00068  *     \phi=ATAN2(Q_{PS},I_{PS}),
00069  * \f}
00070  * where \f$I_{PS1},Q_{PS1}\f$ are the inphase and quadrature prompt correlator outputs respectively.
       The output is in [radians].
00071  */
00072 double pll_four_quadrant_atan(gr_complex prompt_s1);
00073
00074
00075 /*! \brief PLL Costas loop two quadrant arctan discriminator
00076  *
00077  * PLL Costas loop two quadrant arctan discriminator:
00078  * \f{equation}
00079  *     \phi=ATAN\left(\frac{Q_{PS}}{I_{PS}}\right),
00080  * \f}
00081  * where \f$I_{PS1},Q_{PS1}\f$ are the inphase and quadrature prompt correlator outputs respectively.
       The output is in [radians].
00082  */
00083 double pll_cloop_two_quadrant_atan(gr_complex prompt_s1);
00084
00085
00086 /*! \brief DLL Noncoherent Early minus Late envelope normalized discriminator
00087  *
00088  * DLL Noncoherent Early minus Late envelope normalized discriminator:
00089  * \f{equation}
00090  *     error = \frac{y_{intercept} - \text{slope} * \epsilon}{\text{slope}} \frac{E-L}{E+L},
00091  * \f}
00092  * where \f$E=\sqrt{I_{ES}^2+Q_{ES}^2}\f$ is the Early correlator output absolute value and
00093  * \f$L=\sqrt{I_{LS}^2+Q_{LS}^2}\f$ is the Late correlator output absolute value. The output is in
       [chips].
00094  */
00095 double dll_nc_e_minus_l_normalized(gr_complex early_s1, gr_complex late_s1, float spc = 0.5, float
       slope = 1.0, float y_intercept = 1.0);
00096
00097
00098 /*! \brief DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator
00099  *
00100  * DLL Noncoherent Very Early Minus Late Power (VEMLP) normalized discriminator, using the outputs
00101  * of four correlators, Very Early (VE), Early (E), Late (L) and Very Late (VL):
00102  * \f{equation}
00103  *  error=\frac{E-L}{E+L},
00104  * \f}
00105  * where \f$E=\sqrt{I_{VE}^2+Q_{VE}^2+I_{E}^2+Q_{E}^2}\f$ and
00106  * \f$L=\sqrt{I_{VL}^2+Q_{VL}^2+I_{L}^2+Q_{L}^2}\f$ . The output is in [chips].
00107  */
00108 double dll_nc_vemlp_normalized(gr_complex very_early_s1, gr_complex early_s1, gr_complex late_s1,
       gr_complex very_late_s1);
00109
00110
00111 template <typename Fun>
00112 double CalculateSlope(Fun &&f, double x)
00113 {
00114     static constexpr double dx = 1e-6;
00115
00116     return (f(x + dx / 2.0) - f(x - dx / 2.0)) / dx;
00117 }
```

```
00118
00119 template <typename Fun>
00120 double CalculateSlopeAbs(Fun &&f, double x)
00121 {
00122     static constexpr double dx = 1e-6;
00123
00124     return (std::abs(f(x + dx / 2.0)) - std::abs(f(x - dx / 2.0))) / dx;
00125 }
00126
00127 template <typename Fun>
00128 double GetYIntercept(Fun &&f, double x)
00129 {
00130     double slope = CalculateSlope(f, x);
00131     double y1 = f(x);
00132
00133     return y1 - slope * x;
00134 }
00135
00136 template <typename Fun>
00137 double GetYInterceptAbs(Fun &&f, double x)
00138 {
00139     double slope = CalculateSlopeAbs(f, x);
00140     double y1 = std::abs(f(x));
00141     return y1 - slope * x;
00142 }
00143
00144 // SinBocCorrelationFunction and CosBocCorrelationFunction from
00145 // Sousa, F. and Nunes, F., "New Expressions for the Autocorrelation
00146 // Function of BOC GNSS Signals", NAVIGATION - Journal of the Institute
00147 // of Navigation, March 2013.
00148 //
00149 template <int M = 1, int N = M>
00150 double SinBocCorrelationFunction(double offset_in_chips)
00151 {
00152     static constexpr int TWO_P = 2 * M / N;
00153
00154     double abs_tau = std::abs(offset_in_chips);
00155
00156     if (abs_tau > 1.0)
00157         {
00158             return 0.0;
00159         }
00160
00161     int k = static_cast<int>(std::ceil(TWO_P * abs_tau));
00162
00163     double sgn = ((k & 0x01) == 0 ? 1.0 : -1.0);  // (-1)^k
00164
00165     return sgn * (2.0 * (k * k - k * TWO_P - k) / TWO_P + 1.0 +
00166                     (2 * TWO_P - 2 * k + 1) * abs_tau);
00167 }
00168
00169
00170 template <int M = 1, int N = M>
00171 double CosBocCorrelationFunction(double offset_in_chips)
00172 {
00173     static constexpr int TWO_P = 2 * M / N;
00174
00175     double abs_tau = std::abs(offset_in_chips);
00176
00177     if (abs_tau > 1.0)
00178         {
00179             return 0.0;
00180         }
00181
00182     int k = static_cast<int>(std::floor(2.0 * TWO_P * abs_tau));
00183
00184     if ((k & 0x01) == 0)  // k is even
00185         {
00186             double sgn = ((k >> 1) & 0x01 ? -1.0 : 1.0);  // (-1)^(k/2)
00187
00188             return sgn * ((2 * k * TWO_P + 2 * TWO_P - k * k) / (2.0 * TWO_P) + (-2 * TWO_P + k - 1) *
      abs_tau);
00189         }
00190     else
00191         {
00192             double sgn = (((k + 1) >> 1) & 0x01 ? -1.0 : 1.0);  // (-1)^((k+1)/2)
00193
00194             return sgn * ((k * k + 2 * k - 2 * k * TWO_P + 1) / (2.0 * TWO_P) + (2 * TWO_P - k - 2) *
      abs_tau);
00195         }
00196 }
00197
00198
00199 /** \} */
00200 /** \} */
00201 #endif  // GNSS_SDR_TRACKING_DISCRIMINATORS_H
```

13.539 tracking_FLL_PLL_filter.h File Reference

Interface of a hybrid FLL and PLL filter for tracking carrier loop.

Classes

- class Tracking_FLL_PLL_filter

  This class implements a hybrid FLL and PLL filter for tracking carrier loop.

13.539.1 Detailed Description

Interface of a hybrid FLL and PLL filter for tracking carrier loop.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file tracking_FLL_PLL_filter.h.

13.540 tracking_FLL_PLL_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_FLL_PLL_filter.h
00003  * \brief Interface of a hybrid FLL and PLL filter for tracking carrier loop
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_TRACKING_FLL_PLL_FILTER_H
00018 #define GNSS_SDR_TRACKING_FLL_PLL_FILTER_H
00019
00020 /** \addtogroup Tracking
00021  * \{ */
00022 /** \addtogroup Tracking_libs
00023  * \{ */
00024
00025
00026 /*!
00027  * \brief This class implements a hybrid FLL and PLL filter for tracking carrier loop
00028  */
00029 class Tracking_FLL_PLL_filter
00030 {
00031 public:
00032     Tracking_FLL_PLL_filter() = default;
00033     ~Tracking_FLL_PLL_filter() = default;
00034     void set_params(float fll_bw_hz, float pll_bw_hz, int order);
00035     void initialize(float d_acq_carrier_doppler_hz);
00036     float get_carrier_error(float FLL_discriminator, float PLL_discriminator, float
      correlation_time_s);
00037
00038 private:
00039     // FLL + PLL filter parameters
00040     float d_pll_w{0.0};
00041     float d_pll_w0p3{0.0};
00042     float d_pll_w0f2{0.0};
00043     float d_pll_x{0.0};
00044     float d_pll_a2{0.0};
00045     float d_pll_w0f{0.0};
00046     float d_pll_a3{0.0};
00047     float d_pll_w0p2{0.0};
00048     float d_pll_b3{0.0};
00049     float d_pll_w0p{0.0};
00050     int d_order{0};
00051 };
00052
```

```
00053
00054 /** \} */
00055 /** \} */
00056 #endif
```

13.541 tracking_loop_filter.h File Reference

Generic 1st to 3rd order loop filter implementation.

```
#include <vector>
```

Classes

- class Tracking_loop_filter

  This class implements a generic 1st, 2nd or 3rd order loop filter.

13.541.1 Detailed Description

Generic 1st to 3rd order loop filter implementation.

Author

Cillian O'Driscoll, 2015. cillian.odriscoll(at)gmail.com

Class implementing a generic 1st, 2nd or 3rd order loop filter. Based on the bilinear transform of the standard Wiener filter.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tracking_loop_filter.h.

13.542 tracking_loop_filter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_loop_filter.h
00003  * \brief Generic 1st to 3rd order loop filter implementation
00004  * \author Cillian O'Driscoll, 2015. cillian.odriscoll(at)gmail.com
00005  *
00006  * Class implementing a generic 1st, 2nd or 3rd order loop filter. Based
00007  * on the bilinear transform of the standard Wiener filter.
00008  *
00009  * -----------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_TRACKING_LOOP_FILTER_H
00021 #define GNSS_SDR_TRACKING_LOOP_FILTER_H
00022
00023 #include <vector>
00024
00025 /** \addtogroup Tracking
00026  * \{ */
00027 /** \addtogroup Tracking_libs
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief This class implements a generic 1st, 2nd or 3rd order loop filter
00033  *
00034  */
00035 class Tracking_loop_filter
00036 {
00037 public:
00038     Tracking_loop_filter();
00039     ~Tracking_loop_filter() = default;
```

```
00040
00041     Tracking_loop_filter(float update_interval, float noise_bandwidth,
00042         int loop_order = 2,
00043         bool include_last_integrator = false);
00044
00045     Tracking_loop_filter(Tracking_loop_filter&&) = default;                      //!< Move operator
00046     Tracking_loop_filter& operator=(Tracking_loop_filter&& /*other*/) = default;  //!< Move assignment
      operator
00047
00048     float get_noise_bandwidth() const;
00049     float get_update_interval() const;
00050     bool get_include_last_integrator() const;
00051     int get_order() const;
00052
00053     void set_noise_bandwidth(float noise_bandwidth);
00054     void set_update_interval(float update_interval);
00055     void set_include_last_integrator(bool include_last_integrator);
00056     void set_order(int loop_order);
00057
00058     void initialize(float initial_output = 0.0);
00059     float apply(float current_input);
00060
00061 private:
00062     // Compute the filter coefficients:
00063     void update_coefficients();
00064
00065     // Store the last inputs and outputs:
00066     std::vector<float> d_inputs;
00067     std::vector<float> d_outputs;
00068
00069     // Store the filter coefficients:
00070     std::vector<float> d_input_coefficients;
00071     std::vector<float> d_output_coefficients;
00072
00073     // The noise bandwidth (in Hz)
00074     // Note this is an approximation only valid when the product of this
00075     // number and the update interval (T) is small.
00076     float d_noise_bandwidth;
00077
00078     // Loop update interval
00079     float d_update_interval;
00080
00081     // The loop order:
00082     int d_loop_order;
00083
00084     // The current index in the i/o arrays:
00085     int d_current_index;
00086
00087     // Should the last integrator be included?
00088     bool d_include_last_integrator;
00089 };
00090
00091
00092 /** \} */
00093 /** \} */
00094 #endif  // GNSS_SDR_TRACKING_LOOP_FILTER_H
```

13.543 acquisition_interface.h File Reference

Header file of the interface to an acquisition GNSS block.

```
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include <memory>
```

Classes

- class AcquisitionInterface

  This abstract class represents an interface to an acquisition GNSS block.

13.543.1 Detailed Description

Header file of the interface to an acquisition GNSS block.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com Luis Esteve, 2011. luis(at)epsilon-formacion.com

This header file contains the interface to an abstract class for acquisition algorithms. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file acquisition_interface.h.

13.544 acquisition_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file acquisition_interface.h
00003  * \brief Header file of the interface to an acquisition GNSS block.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *         Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *
00007  * This header file contains the interface to an abstract class
00008  * for acquisition algorithms. Since all its methods are virtual,
00009  * this class cannot be instantiated directly, and a subclass can only be
00010  * instantiated directly if all inherited pure virtual methods have been
00011  * implemented by that class or a parent class.
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_ACQUISITION_INTERFACE_H
00025 #define GNSS_SDR_ACQUISITION_INTERFACE_H
00026
00027 #include "gnss_block_interface.h"
00028 #include "gnss_synchro.h"
00029 #include <memory>
00030
00031 /** \addtogroup Core
00032  * \{ */
00033 /** \addtogroup GNSS_Block_Interfaces GNSS block interfaces
00034  * GNSS block interfaces.
00035  * \{ */
00036
00037
00038 template <typename Data>
00039 class Concurrent_Queue;
00040
00041 class ChannelFsm;
00042
00043 /*! \brief This abstract class represents an interface to an acquisition GNSS block.
00044  *
00045  * Abstract class for acquisition algorithms. Since all its methods are virtual,
00046  * this class cannot be instantiated directly, and a subclass can only be
00047  * instantiated directly if all inherited pure virtual methods have been
00048  * implemented by that class or a parent class.
00049  */
00050 class AcquisitionInterface : public GNSSBlockInterface
00051 {
00052 public:
00053     virtual void set_gnss_synchro(Gnss_Synchro* gnss_synchro) = 0;
00054     virtual void set_channel(unsigned int channel_id) = 0;
00055     virtual void set_channel_fsm(std::weak_ptr<ChannelFsm> channel_fsm) = 0;
00056     virtual void set_doppler_center(int /*doppler_center*/) {}
00057     //! Number of Doppler bins to search, centered on set_doppler_center().
00058     //! doppler_num_bins == 0 searches the full configured Doppler range
00059     //! (computed from doppler_max/doppler_step -- the one value a caller
00060     //! can't supply directly, since that count is implementation-private);
00061     //! any other value is the literal candidate bin count to search -- 1 for
00062     //! an exactly-known Doppler (as when assisted by an already-tracked
00063     //! primary frequency or a visibility-aware prediction), or any N in
00064     //! between for a search with partial uncertainty.
00065     virtual void set_doppler_num_bins(unsigned int /*doppler_num_bins*/) {}
00066     virtual void set_local_code() = 0;
00067     virtual signed int mag() = 0;
00068     virtual void reset() = 0;
```

```
00069     virtual void stop_acquisition() = 0;
00070     virtual void set_resampler_latency(uint32_t latency_samples) = 0;
00071 };
00072
00073
00074 /** \} */
00075 /** \} */
00076 #endif  // GNSS_SDR_ACQUISITION_INTERFACE */
```

13.545 channel_interface.h File Reference

This class represents an interface to a channel GNSS block.

```
#include "gnss_block_interface.h"
#include "gnss_signal.h"
#include <cstdint>
```

Classes

- class ChannelInterface

  This abstract class represents an interface to a channel GNSS block.

13.545.1 Detailed Description

This class represents an interface to a channel GNSS block.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com Luis Esteve, 2011. luis(at)epsilon-formacion.com

Abstract class for channel blocks. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file channel_interface.h.

13.546 channel_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel_interface.h
00003  * \brief This class represents an interface to a channel GNSS block.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *         Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *
00007  * Abstract class for channel blocks. Since all its methods are virtual,
00008  * this class cannot be instantiated directly, and a subclass can only be
00009  * instantiated directly if all inherited pure virtual methods have been
00010  * implemented by that class or a parent class.
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_CHANNEL_INTERFACE_H
00024 #define GNSS_SDR_CHANNEL_INTERFACE_H
00025
00026 #include "gnss_block_interface.h"
00027 #include "gnss_signal.h"
00028 #include <cstdint>
00029
00030 /** \addtogroup Core
00031  * \{ */
```

```
00032 /** \addtogroup GNSS_Block_Interfaces
00033  * \{ */
00034
00035
00036 /*!
00037  * \brief This abstract class represents an interface to a channel GNSS block.
00038  *
00039  * Abstract class for channel blocks. Since all its methods are pure virtual,
00040  * this class cannot be instantiated directly, and a subclass can only be
00041  * instantiated directly if all inherited pure virtual methods have been
00042  * implemented by that class or a parent class.
00043  */
00044 class ChannelInterface : public GNSSBlockInterface
00045 {
00046 public:
00047     virtual gr::basic_block_sptr get_left_block_trk() = 0;
00048     virtual gr::basic_block_sptr get_right_block_trk() = 0;
00049     virtual gr::basic_block_sptr get_left_block_acq() = 0;
00050     virtual gr::basic_block_sptr get_right_block_acq() = 0;
00051     virtual gr::basic_block_sptr get_left_block() = 0;
00052     virtual gr::basic_block_sptr get_right_block() = 0;
00053     virtual Gnss_Signal get_signal() = 0;
00054     virtual void start_acquisition() = 0;
00055     // doppler_num_bins: number of Doppler bins to search, or 0 (the default)
00056     // for the full configured range -- see AcquisitionInterface::set_doppler_num_bins().
00057     virtual void assist_acquisition_doppler(double Carrier_Doppler_hz, uint32_t doppler_num_bins = 0)
      = 0;
00058     virtual void stop_channel() = 0;
00059     virtual void set_signal(const Gnss_Signal&) = 0;
00060 };
00061
00062
00063 /** \} */
00064 /** \} */
00065 #endif  // GNSS_SDR_CHANNEL_INTERFACE_H
```

13.547 configuration_interface.h File Reference

This class represents an interface to configuration parameters.

```
#include <cstdint>
#include <string>
```

Classes

- class ConfigurationInterface

  This abstract class represents an interface to configuration parameters.

13.547.1 Detailed Description

This class represents an interface to configuration parameters.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

The interface defines an accessor method that gets a parameter name as input and returns the value of this parameter, a string, as output. Property names are defined here. This is an abstract class for interfaces.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file configuration_interface.h.

13.548 configuration_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file configuration_interface.h
00003  * \brief This class represents an interface to configuration parameters.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * The interface defines an accessor method that gets a parameter name as input
00007  * and returns the value of this parameter, a string, as output.
```

```
00008  * Property names are defined here. This is an abstract class for interfaces.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CONFIGURATION_INTERFACE_H
00022 #define GNSS_SDR_CONFIGURATION_INTERFACE_H
00023
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup GNSS_Block_Interfaces
00030  * \{ */
00031
00032
00033 /*!
00034  * \brief This abstract class represents an interface to configuration parameters.
00035  *
00036  * The interface defines an accessor method that gets a parameter name as input
00037  * and returns the value of this parameter, a string, as output.
00038  * Property names are defined here. This is an abstract class for interfaces.
00039  * Since all its methods are virtual,
00040  * this class cannot be instantiated directly, and a subclass can only be
00041  * instantiated directly if all inherited pure virtual methods have been
00042  * implemented by that class or a parent class.
00043  */
00044 class ConfigurationInterface
00045 {
00046 public:
00047     virtual ~ConfigurationInterface() = default;
00048     virtual std::string property(std::string property_name, std::string default_value) const = 0;
00049     virtual bool property(std::string property_name, bool default_value) const = 0;
00050     virtual int64_t property(std::string property_name, int64_t default_value) const = 0;
00051     virtual uint64_t property(std::string property_name, uint64_t default_value) const = 0;
00052     virtual int32_t property(std::string property_name, int32_t default_value) const = 0;
00053     virtual uint32_t property(std::string property_name, uint32_t default_value) const = 0;
00054     virtual int16_t property(std::string property_name, int16_t default_value) const = 0;
00055     virtual uint16_t property(std::string property_name, uint16_t default_value) const = 0;
00056     virtual float property(std::string property_name, float default_value) const = 0;
00057     virtual double property(std::string property_name, double default_value) const = 0;
00058     virtual void set_property(std::string property_name, std::string value) = 0;
00059     virtual bool is_present(const std::string& property_name) const = 0;
00060 };
00061
00062
00063 /** \} */
00064 /** \} */
00065 #endif  // GNSS_SDR_CONFIGURATION_INTERFACE_H
```

13.549 gnss_block_interface.h File Reference

This interface represents a GNSS block.

```
#include <gnuradio/top_block.h>
#include <cassert>
#include <string>
#include <utility>
#include <boost/make_shared.hpp>
#include <boost/shared_ptr.hpp>
```

Classes

- class GNSSBlockInterface

  This abstract class represents an interface to GNSS blocks.

Typedefs

- template<typename T>
  using gnss_shared_ptr = boost::shared_ptr<T>

Functions

- template<typename C, typename... Args>
  gnss_shared_ptr< C > gnss_make_shared (Args &&... args)

13.549.1 Detailed Description

This interface represents a GNSS block.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

Abstract class for GNSS block interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_block_interface.h.

13.550 gnss_block_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_block_interface.h
00003  * \brief This interface represents a GNSS block.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * Abstract class for GNSS block interfaces. Since all its methods are virtual,
00007  * this class cannot be instantiated directly, and a subclass can only be
00008  * instantiated directly if all inherited pure virtual methods have been
00009  * implemented by that class or a parent class.
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022
00023 #ifndef GNSS_SDR_GNSS_BLOCK_INTERFACE_H
00024 #define GNSS_SDR_GNSS_BLOCK_INTERFACE_H
00025
00026 #include <gnuradio/top_block.h>
00027 #include <cassert>
00028 #include <string>
00029 #include <utility>  // for std::forward
00030
00031 /** \addtogroup Core
00032  * \{ */
00033 /** \addtogroup GNSS_Block_Interfaces
00034  * \{ */
00035
00036 // clang-format off
00037 #if GNURADIO_USES_STD_POINTERS
00038 #include <memory>
00039 template <typename T>
00040 using gnss_shared_ptr = std::shared_ptr<T>;
00041 template <typename C, typename... Args>
00042 gnss_shared_ptr<C> gnss_make_shared(Args &&... args)
00043 {
00044     return std::make_shared<C>(std::forward<Args>(args)...);
00045 }
00046 #else
00047 #include <boost/make_shared.hpp>
```

```
00048 #include <boost/shared_ptr.hpp>
00049 template <typename T>
00050 using gnss_shared_ptr = boost::shared_ptr<T>;
00051 template <typename C, typename... Args>
00052 gnss_shared_ptr<C> gnss_make_shared(Args &&... args)
00053 {
00054     return boost::make_shared<C>(std::forward<Args>(args)...);
00055 }
00056 #endif
00057 // clang-format on
00058
00059
00060 /*!
00061  * \brief This abstract class represents an interface to GNSS blocks.
00062  *
00063  * Abstract class for GNSS block interfaces. Since all its methods are virtual,
00064  * this class cannot be instantiated directly, and a subclass can only be
00065  * instantiated directly if all inherited pure virtual methods have been
00066  * implemented by that class or a parent class.
00067  */
00068 class GNSSBlockInterface
00069 {
00070 public:
00071     virtual ~GNSSBlockInterface() = default;
00072     virtual std::string role() = 0;
00073     virtual std::string implementation() = 0;
00074     virtual size_t item_size() = 0;
00075     virtual void connect(gr::top_block_sptr top_block) = 0;
00076     virtual void disconnect(gr::top_block_sptr top_block) = 0;
00077
00078     virtual gr::basic_block_sptr get_left_block() = 0;
00079     virtual gr::basic_block_sptr get_right_block() = 0;
00080
00081     virtual gr::basic_block_sptr get_left_block(int RF_channel)
00082     {
00083         assert(RF_channel >= 0);
00084         if (RF_channel == 0)
00085             {
00086             };  // avoid unused param warning
00087         return nullptr;  // added to support raw array access (non pure virtual to allow left
      unimplemented)= 0;
00088     }
00089     virtual gr::basic_block_sptr get_right_block(int RF_channel)
00090     {
00091         assert(RF_channel >= 0);
00092         if (RF_channel == 0)
00093             {
00094             };  // avoid unused param warning
00095         return nullptr;  // added to support raw array access (non pure virtual to allow left
      unimplemented)= 0;
00096     }
00097
00098     /*!
00099      * \brief Start the flow of samples if needed.
00100      */
00101     virtual void start() {};
00102 };
00103
00104
00105 /** \} */
00106 /** \} */
00107 #endif  // GNSS_SDR_GNSS_BLOCK_INTERFACE_H
```

13.551 observables_interface.h File Reference

This class represents an interface to an Observables block.
`#include "gnss_block_interface.h"`

Classes

- class ObservablesInterface

  This abstract class represents an interface to an observables block.

13.551.1 Detailed Description

This class represents an interface to an Observables block.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Abstract class for Observables modules. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file observables_interface.h.

13.552 observables_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file observables_interface.h
00003  * \brief This class represents an interface to an Observables block.
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * Abstract class for Observables modules. Since all its methods are virtual,
00007  * this class cannot be instantiated directly, and a subclass can only be
00008  * instantiated directly if all inherited pure virtual methods have been
00009  * implemented by that class or a parent class.
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022
00023 #ifndef GNSS_SDR_OBSERVABLES_INTERFACE_H
00024 #define GNSS_SDR_OBSERVABLES_INTERFACE_H
00025
00026 #include "gnss_block_interface.h"
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup GNSS_Block_Interfaces
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This abstract class represents an interface to an observables block.
00036  *
00037  * Abstract class for pseudorange_intefaces, derived from GNSSBlockInterface.
00038  * Since all its methods are virtual,
00039  * this class cannot be instantiated directly, and a subclass can only be
00040  * instantiated directly if all inherited pure virtual methods have been
00041  * implemented by that class or a parent class.
00042  */
00043 class ObservablesInterface : public GNSSBlockInterface
00044 {
00045 public:
00046     virtual void reset() = 0;
00047 };
00048
00049
00050 /** \} */
00051 /** \} */
00052 #endif  // GNSS_SDR_OBSERVABLES_INTERFACE_H
```

13.553 pvt_interface.h File Reference

This class represents an interface to a PVT block.

```
#include "beidou_cnav1_ephemeris.h"
#include "beidou_dnav_almanac.h"
#include "beidou_dnav_ephemeris.h"
#include "galileo_almanac.h"
#include "galileo_ephemeris.h"
```

```
#include "glonass_gnav_almanac.h"
#include "glonass_gnav_ephemeris.h"
#include "glonass_gnav_utc_model.h"
#include "gnss_block_interface.h"
#include "gps_almanac.h"
#include "gps_cnav_ephemeris.h"
#include "gps_ephemeris.h"
#include <map>
```

Classes

- class PvtInterface

  This class represents an interface to a PVT block.

13.553.1 Detailed Description

This class represents an interface to a PVT block.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Abstract class for PVT solvers. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file pvt_interface.h.

13.554 pvt_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file pvt_interface.h
00003  * \brief This class represents an interface to a PVT block.
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * Abstract class for PVT solvers. Since all its methods are virtual,
00007  * this class cannot be instantiated directly, and a subclass can only be
00008  * instantiated directly if all inherited pure virtual methods have been
00009  * implemented by that class or a parent class.
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022
00023 #ifndef GNSS_SDR_PVT_INTERFACE_H
00024 #define GNSS_SDR_PVT_INTERFACE_H
00025
00026 #include "beidou_cnav1_ephemeris.h"
00027 #include "beidou_dnav_almanac.h"
00028 #include "beidou_dnav_ephemeris.h"
00029 #include "galileo_almanac.h"
00030 #include "galileo_ephemeris.h"
00031 #include "glonass_gnav_almanac.h"
00032 #include "glonass_gnav_ephemeris.h"
00033 #include "glonass_gnav_utc_model.h"
00034 #include "gnss_block_interface.h"
00035 #include "gps_almanac.h"
00036 #include "gps_cnav_ephemeris.h"
00037 #include "gps_ephemeris.h"
00038 #include <map>
00039
```

```
00040 /** \addtogroup Core
00041  * \{ */
00042 /** \addtogroup GNSS_Block_Interfaces
00043  * \{ */
00044
00045
00046 /*!
00047  * \brief This class represents an interface to a PVT block.
00048  *
00049  * Abstract class for PVT interfaces, derived from GNSSBlockInterface.
00050  * Since all its methods are virtual,
00051  * this class cannot be instantiated directly, and a subclass can only be
00052  * instantiated directly if all inherited pure virtual methods have been
00053  * implemented by that class or a parent class.
00054  */
00055 class PvtInterface : public GNSSBlockInterface
00056 {
00057 public:
00058     virtual void reset() = 0;
00059     virtual void clear_ephemeris() = 0;
00060     virtual std::map<int, Gps_CNAV_Ephemeris> get_gps_cnav_ephemeris() const = 0;
00061     virtual std::map<int, Glonass_Gnav_Ephemeris> get_glonass_ephemeris() const = 0;
00062     virtual std::map<int, Glonass_Gnav_Almanac> get_glonass_almanac() const = 0;
00063     virtual Glonass_Gnav_Utc_Model get_glonass_utc_model() const = 0;
00064     // Warm start: drops the ephemeris only and keeps the almanac already in
00065     // memory (almanac current, ephemeris unknown or stale). Unlike a
00066     // clear_ephemeris() followed by an XML reload, this cannot replace a live,
00067     // fresher almanac with a stale snapshot from disk.
00068     virtual void clear_ephemeris_keep_almanac() = 0;
00069     virtual std::map<int, Gps_Ephemeris> get_gps_ephemeris() const = 0;
00070     virtual std::map<int, Galileo_Ephemeris> get_galileo_ephemeris() const = 0;
00071     virtual std::map<int, Beidou_Dnav_Ephemeris> get_beidou_dnav_ephemeris() const = 0;
00072     // Optional navigation families: implementations without B-CNAV support
00073     // retain the existing DNAV/almanac visibility path.
00074     virtual std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav1_ephemeris() const { return {}; }
00075     virtual std::map<int, Beidou_Cnav1_Ephemeris> get_beidou_cnav2_ephemeris() const { return {}; }
00076     virtual std::map<int, Gps_Almanac> get_gps_almanac() const = 0;
00077     virtual std::map<int, Galileo_Almanac> get_galileo_almanac() const = 0;
00078     virtual std::map<int, Beidou_Dnav_Almanac> get_beidou_dnav_almanac() const = 0;
00079
00080     virtual bool get_latest_PVT(double* longitude_deg,
00081         double* latitude_deg,
00082         double* height_m,
00083         double* ground_speed_kmh,
00084         double* course_over_ground_deg,
00085         time_t* UTC_time) = 0;
00086 };
00087
00088
00089 /** \} */
00090 /** \} */
00091 #endif  // GNSS_SDR_PVT_INTERFACE_H
```

13.555 signal_source_interface.h

```
00001 /*!
00002  * \signal_source_interface.h
00003  * \brief Header file of the interface to a signal_source GNSS block.
00004  * \author Jim Melton, 2020. jim.melton(at)sncorp.com
00005  *
00006  * This header file contains the interface to an abstract class for
00007  * signal sources. Since all its methods are virtual, this class
00008  * cannot be instantiated directly, and a subclass can only be
00009  * instantiated directly if all inherited pure virtual methods have
00010  * been implemented by that class or a parent class.
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  *
00016  * GNSS-SDR is a software defined Global Navigation
00017  *          Satellite Systems receiver
00018  *
00019  * This file is part of GNSS-SDR.
00020  *
00021  * SPDX-License-Identifier: GPL-3.0-or-later
00022  *
00023  * -----------------------------------------------------------------------------
00024  */
00025
00026 #ifndef GNSS_SDR_SIGNAL_SOURCE_INTERFACE_H
00027 #define GNSS_SDR_SIGNAL_SOURCE_INTERFACE_H
00028
00029 #include "gnss_block_interface.h"
```

```
00030
00031 #if USE_GLOG_AND_GFLAGS
00032 #include <glog/logging.h>
00033 #else
00034 #include <absl/log/log.h>
00035 #endif
00036
00037 /** \addtogroup Core
00038  * \{ */
00039 /** \addtogroup GNSS_Block_Interfaces GNSS block interfaces
00040  * GNSS block interfaces.
00041  * \{ */
00042
00043 /*! \brief This abstract class represents an interface to signal_source GNSS block.
00044  *
00045  * Abstract class for signal sources. Since all its methods are virtual,
00046  * this class cannot be instantiated directly, and a subclass can only be
00047  * instantiated directly if all inherited pure virtual methods have been
00048  * implemented by that class or a parent class.
00049  */
00050
00051 class SignalSourceInterface : public GNSSBlockInterface
00052 {
00053 public:
00054     virtual size_t getRfChannels() const = 0;
00055
00056 protected:
00057     SignalSourceInterface()
00058     {
00059         VLOG(1) « "SignalSourceInterface: " « this « " ctor";
00060     }
00061
00062 public:  // required for polymorphic destruction
00063     ~SignalSourceInterface()
00064     {
00065         VLOG(1) « "SignalSourceInterface: " « this « " dtor";
00066     }
00067 };
00068
00069
00070 #endif
```

13.556 telemetry_decoder_interface.h File Reference

This class represents an interface to a telemetry decoder block.

```
#include "gnss_block_interface.h"
#include "gnss_satellite.h"
```

Classes

- class TelemetryDecoderInterface

  This abstract class represents an interface to a navigation GNSS block.

13.556.1 Detailed Description

This class represents an interface to a telemetry decoder block.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

Abstract class for telemetry decoders. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file telemetry_decoder_interface.h.

13.557 telemetry_decoder_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file telemetry_decoder_interface.h
00003  * \brief This class represents an interface to a telemetry decoder block.
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * Abstract class for telemetry decoders. Since all its methods are virtual,
00007  * this class cannot be instantiated directly, and a subclass can only be
00008  * instantiated directly if all inherited pure virtual methods have been
00009  * implemented by that class or a parent class.
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022
00023 #ifndef GNSS_SDR_TELEMETRY_DECODER_INTERFACE_H
00024 #define GNSS_SDR_TELEMETRY_DECODER_INTERFACE_H
00025
00026 #include "gnss_block_interface.h"
00027 #include "gnss_satellite.h"
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup GNSS_Block_Interfaces
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief This abstract class represents an interface to a navigation GNSS block.
00037  *
00038  * Abstract class for navigation interfaces. Since all its methods are virtual,
00039  * this class cannot be instantiated directly, and a subclass can only be
00040  * instantiated directly if all inherited pure virtual methods have been
00041  * implemented by that class or a parent class.
00042  */
00043 class TelemetryDecoderInterface : public GNSSBlockInterface
00044 {
00045 public:
00046     virtual void reset() = 0;
00047     virtual void set_satellite(const Gnss_Satellite& sat) = 0;
00048     virtual void set_channel(int channel) = 0;
00049 };
00050
00051
00052 /** \} */
00053 /** \} */
00054 #endif  // GNSS_SDR_TELEMETRY_DECODER_INTERFACE_H
```

13.558 tracking_interface.h File Reference

This class represents an interface to a tracking block.

```
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
```

Classes

- class TrackingInterface

  This abstract class represents an interface to a tracking block.

13.558.1 Detailed Description

This class represents an interface to a tracking block.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

Abstract class for tracking interfaces. Since all its methods are virtual, this class cannot be instantiated directly, and a subclass can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tracking_interface.h.

13.559 tracking_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tracking_interface.h
00003  * \brief This class represents an interface to a tracking block.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * Abstract class for tracking interfaces. Since all its methods are virtual,
00007  * this class cannot be instantiated directly, and a subclass can only be
00008  * instantiated directly if all inherited pure virtual methods have been
00009  * implemented by that class or a parent class.
00010  *
00011  * -----------------------------------------------------------------------------
00012  *
00013  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00014  * This file is part of GNSS-SDR.
00015  *
00016  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00017  * SPDX-License-Identifier: GPL-3.0-or-later
00018  *
00019  * -----------------------------------------------------------------------------
00020  */
00021
00022
00023 #ifndef GNSS_SDR_TRACKING_INTERFACE_H
00024 #define GNSS_SDR_TRACKING_INTERFACE_H
00025
00026 #include "gnss_block_interface.h"
00027 #include "gnss_synchro.h"
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup GNSS_Block_Interfaces
00032  * \{ */
00033
00034
00035 template <typename Data>
00036 class Concurrent_Queue;
00037
00038 /*!
00039  * \brief This abstract class represents an interface to a tracking block.
00040  *
00041  * Abstract class for tracking interfaces. Since all its methods are virtual,
00042  * this class cannot be instantiated directly, and a subclass can only be
00043  * instantiated directly if all inherited pure virtual methods have been
00044  * implemented by that class or a parent class.
00045  *
00046  */
00047 class TrackingInterface : public GNSSBlockInterface
00048 {
00049 public:
00050     virtual void start_tracking() = 0;
00051     virtual void stop_tracking() = 0;
00052     virtual void set_gnss_synchro(Gnss_Synchro* gnss_synchro) = 0;
00053     virtual void set_channel(unsigned int channel) = 0;
00054 };
00055
00056
00057 /** \} */
00058 /** \} */
00059 #endif  // GNSS_SDR_TRACKING_INTERFACE_H
```

13.560 channel_event.h File Reference

Class that defines a channel event.

```
#include <memory>
```

Classes

- class Channel_Event

Typedefs

- using channel_event_sptr = std::shared_ptr<Channel_Event>

Functions

- channel_event_sptr **channel_event_make** (int channel_id, int event_type)

13.560.1 Detailed Description

Class that defines a channel event.

Author

Javier Arribas, 2019. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file channel_event.h.

13.561 channel_event.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel_event.h
00003  * \brief Class that defines a channel event
00004  * \author Javier Arribas, 2019. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_CHANNEL_EVENT_H
00018 #define GNSS_SDR_CHANNEL_EVENT_H
00019
00020 #include <memory>
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup Core_Receiver_Library core_libs
00025  * \{ */
00026
00027
00028 class Channel_Event;
00029
00030 using channel_event_sptr = std::shared_ptr<Channel_Event>;
00031
00032 channel_event_sptr channel_event_make(int channel_id, int event_type);
00033
00034 class Channel_Event
00035 {
00036 public:
00037     int channel_id;
00038     int event_type;
00039
00040 private:
00041     friend channel_event_sptr channel_event_make(int channel_id, int event_type);
00042     Channel_Event(int channel_id_, int event_type_);
00043 };
00044
```

```
00045
00046 /** \} */
00047 /** \} */
00048 #endif  // GNSS_SDR_CHANNEL_EVENT_H
```

13.562 channel_status_msg_receiver.h File Reference

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

```
#include "gnss_block_interface.h"
#include "gnss_synchro.h"
#include "monitor_pvt.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <map>
#include <memory>
```

Classes

- class channel_status_msg_receiver

  GNU Radio block that receives asynchronous channel messages from tlm blocks.

Typedefs

- using channel_status_msg_receiver_sptr = gnss_shared_ptr<channel_status_msg_receiver>

Functions

- channel_status_msg_receiver_sptr **channel_status_msg_receiver_make** ()

13.562.1 Detailed Description

GNU Radio block that receives asynchronous channel messages from acquisition and tracking blocks.

Author

Javier Arribas, 2019. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file channel_status_msg_receiver.h.

13.563 channel_status_msg_receiver.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file channel_status_msg_receiver.h
00003  * \brief GNU Radio block that receives asynchronous channel messages from
00004  * acquisition and tracking blocks
00005  * \author Javier Arribas, 2019. jarribas(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_CHANNEL_STATUS_MSG_RECEIVER_CC_H
00019 #define GNSS_SDR_CHANNEL_STATUS_MSG_RECEIVER_CC_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "gnss_synchro.h"
```

```
00023 #include "monitor_pvt.h"
00024 #include <gnuradio/block.h>
00025 #include <pmt/pmt.h>
00026 #include <map>
00027 #include <memory>
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup Core_Receiver_Library
00032  * \{ */
00033
00034
00035 class channel_status_msg_receiver;
00036
00037 using channel_status_msg_receiver_sptr = gnss_shared_ptr<channel_status_msg_receiver>;
00038
00039 channel_status_msg_receiver_sptr channel_status_msg_receiver_make();
00040
00041 /*!
00042  * \brief GNU Radio block that receives asynchronous channel messages from tlm blocks
00043  */
00044 class channel_status_msg_receiver : public gr::block
00045 {
00046 public:
00047     ~channel_status_msg_receiver() = default;  //!< Default destructor
00048
00049     /*!
00050      * \brief return the current status map of all channels with valid telemetry
00051      */
00052     std::map<int, std::shared_ptr<Gnss_Synchro» get_current_status_map();
00053
00054     /*!
00055      * \brief return the current status map of all locked channels, including
00056      * those without a valid time reference yet (no valid pseudorange). Meant
00057      * for the secondary-frequency acquisition assistance, which only needs the
00058      * satellite and its Doppler.
00059      */
00060     std::map<int, std::shared_ptr<Gnss_Synchro» get_current_tracking_map();
00061
00062     /*!
00063      * \brief return the latest successful PVT and optionally elapsed sample time
00064      */
00065     Monitor_Pvt get_current_status_pvt(double* receiver_time_s = nullptr);
00066
00067     /*!
00068      * \brief forget the status of a channel that has been stopped, so that a
00069      * stale entry is not reported until the channel delivers new observables
00070      */
00071     void clear_channel_status(int channel_id);
00072
00073 private:
00074     friend channel_status_msg_receiver_sptr channel_status_msg_receiver_make();
00075     channel_status_msg_receiver();
00076     void msg_handler_channel_status(const pmt::pmt_t& msg);
00077     Monitor_Pvt d_pvt_status{};
00078     double d_receiver_time_s{0.0};  // elapsed sample time, including epochs without a fix
00079     std::map<int, std::shared_ptr<Gnss_Synchro» d_channel_status_map;
00080     std::map<int, std::shared_ptr<Gnss_Synchro» d_channel_tracking_map;
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_CHANNEL_STATUS_MSG_RECEIVER_CC_H
```

13.564 command_event.h File Reference

Class that defines a receiver command event.

`#include <memory>`

Classes

- class Command_Event

Typedefs

- using command_event_sptr = std::shared_ptr<Command_Event>

Functions

- command_event_sptr **command_event_make** (int command_id, int event_type)

13.564.1 Detailed Description

Class that defines a receiver command event.

Author

Javier Arribas, 2019. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file command_event.h.

13.565 command_event.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file command_event.h
00003  * \brief Class that defines a receiver command event
00004  * \author Javier Arribas, 2019. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_COMMAND_EVENT_H
00018 #define GNSS_SDR_COMMAND_EVENT_H
00019
00020 #include <memory>
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup Core_Receiver_Library
00025  * \{ */
00026
00027
00028 class Command_Event;
00029
00030 using command_event_sptr = std::shared_ptr<Command_Event>;
00031
00032 command_event_sptr command_event_make(int command_id, int event_type);
00033
00034 class Command_Event
00035 {
00036 public:
00037     int command_id;
00038     int event_type;
00039
00040 private:
00041     friend command_event_sptr command_event_make(int command_id, int event_type);
00042     Command_Event(int command_id_, int event_type_);
00043 };
00044
00045
00046 /** \} */
00047 /** \} */
00048 #endif  // GNSS_SDR_COMMAND_EVENT_H
```

13.566 galileo_e6_has_msg_receiver.h File Reference

GNU Radio block that processes Galileo HAS message pages received from Galileo E6B telemetry blocks. After successful decoding, sends the content to the PVT block.

```
#include "Galileo_CNAV.h"
#include "galileo_has_data.h"
#include "gnss_block_interface.h"
```

```
#include "nav_message_packet.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <bitset>
#include <cstdint>
#include <map>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class galileo_e6_has_msg_receiver

  GNU Radio block that receives asynchronous Galileo HAS message pages from the telemetry blocks, stores them in memory, and decodes HAS messages when enough data have been received. The decoded HAS message is sent to the PVT block.

Typedefs

- using galileo_e6_has_msg_receiver_sptr = gnss_shared_ptr<galileo_e6_has_msg_receiver>

Functions

- galileo_e6_has_msg_receiver_sptr **galileo_e6_has_msg_receiver_make** ()

13.566.1 Detailed Description

GNU Radio block that processes Galileo HAS message pages received from Galileo E6B telemetry blocks. After successful decoding, sends the content to the PVT block.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

Javier Arribas, 2021. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_e6_has_msg_receiver.h.

13.567 galileo_e6_has_msg_receiver.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_e6_has_msg_receiver.h
00003  * \brief GNU Radio block that processes Galileo HAS message pages received from
00004  * Galileo E6B telemetry blocks. After successful decoding, sends the content to
00005  * the PVT block.
00006  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00007  * \author Javier Arribas, 2021. jarribas(at)cttc.es
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_GALILEO_E6_HAS_MSG_RECEIVER_H
00021 #define GNSS_SDR_GALILEO_E6_HAS_MSG_RECEIVER_H
00022
00023 #include "Galileo_CNAV.h"          // for GALILEO_CNAV_*  constants
```

```
00024 #include "galileo_has_data.h"      // for Galileo_HAS_data
00025 #include "gnss_block_interface.h"  // for gnss_shared_ptr
00026 #include "nav_message_packet.h"    // for Nav_Message_Packet
00027 #include <gnuradio/block.h>        // for gr::block
00028 #include <pmt/pmt.h>               // for pmt::pmt_t
00029 #include <bitset>
00030 #include <cstdint>
00031 #include <map>
00032 #include <memory>  // for std::unique_ptr
00033 #include <string>
00034 #include <utility>  // std::pair
00035 #include <vector>
00036
00037 /** \addtogroup Core
00038  * \{ */
00039 /** \addtogroup Core_Receiver_Library
00040  * \{ */
00041
00042 class Galileo_HAS_page;
00043 class ReedSolomon;
00044 class galileo_e6_has_msg_receiver;
00045
00046 using galileo_e6_has_msg_receiver_sptr = gnss_shared_ptr<galileo_e6_has_msg_receiver>;
00047
00048 galileo_e6_has_msg_receiver_sptr galileo_e6_has_msg_receiver_make();
00049
00050 /*!
00051  * \brief GNU Radio block that receives asynchronous Galileo HAS message pages
00052  * from the telemetry blocks, stores them in memory, and decodes HAS messages
00053  * when enough data have been received.
00054  * The decoded HAS message is sent to the PVT block.
00055  */
00056 class galileo_e6_has_msg_receiver : public gr::block
00057 {
00058 public:
00059     ~galileo_e6_has_msg_receiver() = default;  //!< Default destructor
00060     void set_enable_navdata_monitor(bool enable);
00061     std::shared_ptr<Galileo_HAS_data> process_test_page(const pmt::pmt_t& msg);  //!< For testing
      purposes only
00062
00063 private:
00064     friend galileo_e6_has_msg_receiver_sptr galileo_e6_has_msg_receiver_make();
00065     galileo_e6_has_msg_receiver();
00066
00067     void msg_handler_galileo_e6_has(const pmt::pmt_t& msg);
00068     void process_HAS_page(const Galileo_HAS_page& has_page);
00069     void clear_HAS_state();
00070     void clear_mask_data(uint8_t mask_id);
00071     void read_MT1_header(const std::string& message_header);
00072     void delete_outdated_data(const Galileo_HAS_page& has_page);
00073     void update_TOW_cache(const Galileo_HAS_page& has_page);
00074     std::shared_ptr<Galileo_HAS_data> get_ready_HAS_data(uint32_t week, uint32_t tow, uint64_t
      time_stamp);
00075     bool read_MT1_body(const std::string& message_body, uint64_t time_stamp);
00076     int decode_message_type1(uint8_t message_id, uint8_t message_size, uint64_t time_stamp);
00077
00078     uint16_t read_has_message_header_parameter_uint16(const std::bitset<GALILEO_CNAV_MT1_HEADER_BITS>&
      bits, const std::pair<int32_t, int32_t>& parameter) const;
00079     uint8_t read_has_message_header_parameter_uint8(const std::bitset<GALILEO_CNAV_MT1_HEADER_BITS>&
      bits, const std::pair<int32_t, int32_t>& parameter) const;
00080     bool read_has_message_header_parameter_bool(const std::bitset<GALILEO_CNAV_MT1_HEADER_BITS>& bits,
      const std::pair<int32_t, int32_t>& parameter) const;
00081     uint32_t get_TOW_for_page(const Galileo_HAS_page& has_page) const;
00082     uint32_t get_week_for_page(const Galileo_HAS_page& has_page, uint32_t tow) const;
00083     bool is_cache_timestamp_valid(uint64_t cache_timestamp, uint64_t current_timestamp) const;
00084     bool is_supported_gnss_id(uint8_t gnss_id) const;
00085     bool is_validity_interval_index_valid(uint8_t validity_interval_index) const;
00086
00087     uint64_t read_has_message_body_uint64(const std::string& bits) const;
00088     uint16_t read_has_message_body_uint16(const std::string& bits) const;
00089     int16_t read_has_message_body_int16(const std::string& bits) const;
00090     uint8_t read_has_message_body_uint8(const std::string& bits) const;
00091
00092     template <class T>
00093     std::string debug_print_vector(const std::string& title, const std::vector<T>& vec) const;  //
      only for debug purposes
00094
00095     template <class T>
00096     std::string debug_print_matrix(const std::string& title, const std::vector<std::vector<T>& mat)
      const;  // only for debug purposes
00097
00098     std::unique_ptr<ReedSolomon> d_rs;
00099     Galileo_HAS_data d_HAS_data{};
00100     Nav_Message_Packet d_nav_msg_packet;
00101
00102     // Store decoding matrices and received PIDs
00103     std::vector<std::vector<uint64_t» d_received_timestamps;
```

```
00104     std::vector<std::vector<std::vector<uint8_t>> d_C_matrix;
00105     std::vector<std::vector<uint8_t> d_M_matrix;
00106     std::vector<std::vector<uint8_t> d_received_pids;
00107     std::vector<uint64_t> d_printed_timestamps;
00108     std::vector<bool> d_printed_mids;
00109
00110     // Store masks
00111     std::vector<int> d_nsat_in_mask_id;
00112     std::vector<std::vector<uint8_t> d_gnss_id_in_mask;
00113     std::vector<std::vector<uint64_t> d_satellite_mask;
00114     std::vector<std::vector<uint16_t> d_signal_mask;
00115     std::vector<std::vector<bool> d_cell_mask_availability_flag;
00116     std::vector<std::vector<std::vector<std::vector<bool>>> d_cell_mask;
00117     std::vector<uint8_t> d_nsys_in_mask;
00118     std::vector<std::vector<uint8_t> d_nav_message_mask;
00119     std::vector<uint64_t> d_mask_timestamps;
00120
00121     std::map<std::pair<uint8_t, uint8_t>, std::vector<uint16_t> d_iod_ref_map;
00122     std::map<std::pair<uint8_t, uint8_t>, uint64_t> d_iod_ref_timestamps;
00123
00124     uint8_t d_current_has_status{};
00125     uint8_t d_current_message_id{};
00126     uint64_t d_last_valid_tow_timestamp;
00127     uint32_t d_last_valid_week;
00128     uint32_t d_last_valid_tow;
00129     bool d_new_message{};
00130     bool d_enable_navdata_monitor{};
00131 };
00132
00133
00134 /** \} */
00135 /** \} */
00136 #endif  // GNSS_SDR_GALILEO_E6_HAS_MSG_RECEIVER_H
```

13.568 galileo_tow_map.h File Reference

GNU Radio block that stores TOW for Galileo channels.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <cstdint>
#include <limits>
#include <map>
#include <utility>
```

Classes

- struct GalileoTowMapEntry
- class galileo_tow_map

Typedefs

- using galileo_tow_map_sptr = gnss_shared_ptr<galileo_tow_map>
- using GalileoTowMap = std::map<uint32_t, GalileoTowMapEntry>
- using GalileoTowMapMessage = std::pair<uint32_t, GalileoTowMapEntry>

Functions

- galileo_tow_map_sptr **galileo_tow_map_make** ()

Variables

- constexpr uint32_t GALILEO_TOW_MAP_WEEK_MS = 604800000U
- constexpr uint32_t GALILEO_TOW_MAP_INVALID_WEEK = std::numeric_limits<uint32_t>::max()
- constexpr uint32_t GALILEO_TOW_MAP_INVALID_TOW_MS = std::numeric_limits<uint32_t>::max()
- constexpr uint64_t GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER = std::numeric_limits<uint64_t>::max()

13.568.1 Detailed Description

GNU Radio block that stores TOW for Galileo channels.

Author

Carles Fernandez-Prades, 2022. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_tow_map.h.

13.569 galileo_tow_map.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_tow_map.h
00003  * \brief GNU Radio block that stores TOW for Galileo channels
00004  * \author Carles Fernandez-Prades, 2022. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GALILEO_TOW_MAP_H
00018 #define GNSS_SDR_GALILEO_TOW_MAP_H
00019
00020 #include "gnss_block_interface.h"  // for gnss_shared_ptr
00021 #include <gnuradio/block.h>        // for gr::block
00022 #include <pmt/pmt.h>               // for pmt::pmt_t
00023 #include <cstdint>
00024 #include <limits>
00025 #include <map>
00026 #include <utility>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup Core_Receiver_Library
00031  * \{ */
00032
00033 class galileo_tow_map;
00034
00035 using galileo_tow_map_sptr = gnss_shared_ptr<galileo_tow_map>;
00036
00037 galileo_tow_map_sptr galileo_tow_map_make();
00038
00039 constexpr uint32_t GALILEO_TOW_MAP_WEEK_MS = 604800000U;
00040 constexpr uint32_t GALILEO_TOW_MAP_INVALID_WEEK = std::numeric_limits<uint32_t>::max();
00041 constexpr uint32_t GALILEO_TOW_MAP_INVALID_TOW_MS = std::numeric_limits<uint32_t>::max();
00042 constexpr uint64_t GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER = std::numeric_limits<uint64_t>::max();
00043
00044 struct GalileoTowMapEntry
00045 {
00046     GalileoTowMapEntry() = default;
00047     GalileoTowMapEntry(uint32_t entry_week, uint32_t entry_tow_ms, uint64_t entry_sample_counter)
00048         : week(entry_week),
00049           tow_ms(entry_tow_ms),
00050           sample_counter(entry_sample_counter)
00051     {
00052     }
00053
00054     uint32_t week{GALILEO_TOW_MAP_INVALID_WEEK};
00055     uint32_t tow_ms{GALILEO_TOW_MAP_INVALID_TOW_MS};
00056     uint64_t sample_counter{GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER};
00057 };
00058
00059 using GalileoTowMap = std::map<uint32_t, GalileoTowMapEntry>;
00060 using GalileoTowMapMessage = std::pair<uint32_t, GalileoTowMapEntry>;
00061
00062 class galileo_tow_map : public gr::block
00063 {
00064 public:
00065     ~galileo_tow_map() = default;  //!< Default destructor
00066
00067 private:
```

```
00068     friend galileo_tow_map_sptr galileo_tow_map_make();
00069     galileo_tow_map();
00070
00071     void msg_handler_galileo_tow_map(const pmt::pmt_t& msg);
00072
00073     GalileoTowMap d_galileo_tow;
00074 };
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_GALILEO_TOW_MAP_H
```

13.570 gnss_crypto.h File Reference

Class for computing cryptographic functions.

```
#include "osnma_crypto_material.h"
#include <cstdint>
#include <string>
#include <vector>
#include <openssl/ec.h>
```

Classes

- class Gnss_Crypto

  Class implementing cryptographic functions for Navigation Message Authentication.

13.570.1 Detailed Description

Class for computing cryptographic functions.

Author

Carles Fernandez, 2023-2024. cfernandez(at)cttc.es Cesare Ghionoiu Martinez, 2023-2024. c.↩ ghionoiu-martinez@tu-braunschweig.de

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_crypto.h.

13.571 gnss_crypto.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_crypto.h
00003  * \brief Class for computing cryptographic functions
00004  * \author Carles Fernandez, 2023-2024. cfernandez(at)cttc.es
00005  *   Cesare Ghionoiu Martinez, 2023-2024. c.ghionoiu-martinez@tu-braunschweig.de
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GNSS_CRYPTO_H
00020 #define GNSS_SDR_GNSS_CRYPTO_H
00021
00022 #include "osnma_crypto_material.h"
00023 #include <cstdint>
00024 #include <string>
00025 #include <vector>
00026 #if USE_GNUTLS_FALLBACK
00027 #include <gnutls/abstract.h>
00028 #include <gnutls/gnutls.h>
```

```
00029 #else  // OpenSSL
00030 #include <openssl/ec.h>
00031 #endif
00032
00033 /** \addtogroup Core
00034  * \{ */
00035 /** \addtogroup Core_Receiver_Library
00036  * \{ */
00037
00038 /*!
00039  * \brief Class implementing cryptographic functions
00040  * for Navigation Message Authentication
00041  */
00042 class Gnss_Crypto
00043 {
00044 public:
00045     Gnss_Crypto();  //!< Default constructor
00046
00047     /*!
00048      * Constructor with a .crt or .pem file for the ECDSA Public Key
00049      * and a XML file for the Merkle Tree root.
00050      * Files can be downloaded by registering at https://www.gsc-europa.eu/
00051      */
00052     Gnss_Crypto(const std::string& certFilePath, const std::string& merkleTreePath);
00053     ~Gnss_Crypto() noexcept;  //!< Default destructor
00054
00055     bool have_public_key() const;  //!< Returns true if the ECDSA Public Key is already loaded
00056     void clear_public_key();       //!< Clears the loaded ECDSA Public Key
00057
00058     /*!
00059      * Stores the ECDSA Public Key in a .pem file, which is read in a following run if the .crt file
      is not found
00060      */
00061     bool store_public_key(const std::string& pubKeyFilePath) const;
00062
00063     bool verify_signature_ecdsa_p256(const std::vector<uint8_t>& message, const std::vector<uint8_t>&
      signature) const;  //!< Verify ECDSA-P256 signature (message in plain hex, signature in raw format)
00064     bool verify_signature_ecdsa_p521(const std::vector<uint8_t>& message, const std::vector<uint8_t>&
      signature) const;  //!< Verify ECDSA-P521 signature (message in plain hex, signature in raw format)
00065
00066     std::vector<uint8_t> compute_SHA_256(const std::vector<uint8_t>& input) const;
      //!< Computes SHA-256 hash
00067     std::vector<uint8_t> compute_SHA3_256(const std::vector<uint8_t>& input) const;
      //!< Computes SHA3-256 hash
00068     std::vector<uint8_t> compute_HMAC_SHA_256(const std::vector<uint8_t>& key, const
      std::vector<uint8_t>& input) const;  //!< Computes HMAC-SHA-256 message authentication code
00069     std::vector<uint8_t> compute_CMAC_AES(const std::vector<uint8_t>& key, const std::vector<uint8_t>&
      input) const;      //!< Computes CMAC-AES message authentication code
00070
00071     std::vector<uint8_t> get_merkle_root() const;            //!< Gets the Merkle Tree root node (\f$
      x_{4,0} \f$)
00072     std::string get_public_key_type() const;                 //!< Gets the ECDSA Public Key type
      (ECDSA P-256 / ECDSA P-521 / Unknown)
00073     std::vector<uint8_t> get_public_key_compressed() const;  //!< Gets the ECDSA Public Key in
      compressed format
00074     std::string get_merkle_tree_hash_function() const;
00075
00076     void set_public_key(const std::vector<uint8_t>& publickey);    //!< Sets the ECDSA Public Key
      (publickey compressed format)
00077     void set_public_key_type(const std::string& public_key_type);  //!< Sets the ECDSA Public Key type
      (ECDSA P-256 / ECDSA P-521)
00078     void set_merkle_root(const std::vector<uint8_t>& v);           //!< Sets the Merkle Tree root node
      x(\f$ x_{4,0} \f$)
00079     void set_merkle_tree_hash_function(const std::string& hash_function);
00080     Osnma_Merkle_Tree_Material read_merkle_xml(const std::string& merkleFilePath);  //!> Reads the XML
      file provided from the GSC OSNMA server
00081
00082 private:
00083     void release_public_key() noexcept;
00084     void readPublicKeyFromPEM(const std::string& pemFilePath);
00085     bool readPublicKeyFromCRT(const std::string& crtFilePath);
00086     bool convert_raw_to_der_ecdsa(const std::vector<uint8_t>& raw_signature, std::vector<uint8_t>&
      der_signature) const;
00087 #if USE_GNUTLS_FALLBACK
00088     void decompress_public_key_secp256r1(const std::vector<uint8_t>& compressed_key,
      std::vector<uint8_t>& x, std::vector<uint8_t>& y) const;
00089     void decompress_public_key_secp521r1(const std::vector<uint8_t>& compressed_key,
      std::vector<uint8_t>& x, std::vector<uint8_t>& y) const;
00090     bool pubkey_copy(gnutls_pubkey_t src, gnutls_pubkey_t* dest);
00091     gnutls_pubkey_t d_PublicKey{};
00092 #else  // OpenSSL
00093 #if USE_OPENSSL_3
00094     bool pubkey_copy(EVP_PKEY* src, EVP_PKEY** dest);
00095     EVP_PKEY* d_PublicKey{};
00096 #else  // OpenSSL 1.x
00097     bool pubkey_copy(EC_KEY* src, EC_KEY** dest);
00098     EC_KEY* d_PublicKey = nullptr;
```

```
00099 #endif
00100 #endif
00101     std::vector<uint8_t> d_x_4_0;
00102     std::string d_PublicKeyType;
00103     std::string d_merkle_tree_hash_function{"SHA-256"};
00104 };
00105
00106 /** \} */
00107 /** \} */
00108
00109 #endif  // GNSS_SDR_GNSS_CRYPTO_H
```

13.572 gnss_sdr_fpga_sample_counter.h File Reference

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

```
#include "gnss_block_interface.h"
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <string>
```

Classes

- class gnss_sdr_fpga_sample_counter

Typedefs

- using gnss_sdr_fpga_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_fpga_sample_counter>

Functions

- gnss_sdr_fpga_sample_counter_sptr **gnss_sdr_make_fpga_sample_counter** (double _fs, int32_t _↩ interval_ms)

13.572.1 Detailed Description

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

Author

Javier Arribas 2018. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_sdr_fpga_sample_counter.h.

13.573 gnss_sdr_fpga_sample_counter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_fpga_sample_counter.h
00003  * \brief Simple block to report the current receiver time based on the output
00004  * of the tracking or telemetry blocks
00005  * \author Javier Arribas 2018. jarribas(at)cttc.es
00006  *
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
```

```
00019 #ifndef GNSS_SDR_GNSS_SDR_FPGA_SAMPLE_COUNTER_H
00020 #define GNSS_SDR_GNSS_SDR_FPGA_SAMPLE_COUNTER_H
00021
00022 #include "gnss_block_interface.h"
00023 #include <gnuradio/block.h>
00024 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00025 #include <cstdint>
00026 #include <string>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup Core_Receiver_Library
00031  * \{ */
00032
00033
00034 class gnss_sdr_fpga_sample_counter;
00035
00036 using gnss_sdr_fpga_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_fpga_sample_counter>;
00037
00038 gnss_sdr_fpga_sample_counter_sptr gnss_sdr_make_fpga_sample_counter(double _fs, int32_t _interval_ms);
00039
00040 class gnss_sdr_fpga_sample_counter : public gr::block
00041 {
00042 public:
00043     ~gnss_sdr_fpga_sample_counter();
00044     int general_work(int noutput_items,
00045         gr_vector_int &ninput_items,
00046         gr_vector_const_void_star &input_items,
00047         gr_vector_void_star &output_items);
00048
00049 private:
00050     const std::string device_name = "counter";  // UIO device name
00051
00052     static const uint32_t FPGA_PAGE_SIZE = 0x1000;         // default page size for the
      multicorrelator memory map
00053     static const uint32_t test_reg_sanity_check = 0x55AA;  // value to check the presence of the test
      register (to detect the hw)
00054
00055     friend gnss_sdr_fpga_sample_counter_sptr gnss_sdr_make_fpga_sample_counter(double _fs, int32_t
      _interval_ms);
00056     gnss_sdr_fpga_sample_counter(double _fs, int32_t _interval_ms);
00057     uint32_t test_register(uint32_t writeval);
00058     void configure_samples_per_output(uint32_t interval);
00059     void close_device(void);
00060     void open_device(void);
00061     bool start();
00062     bool stop();
00063     void wait_for_interrupt(void) const;
00064
00065     volatile uint32_t *map_base;  // driver memory map
00066
00067     double fs;
00068     uint64_t sample_counter;
00069     uint64_t last_sample_counter;
00070     uint64_t current_T_rx_ms;  // Receiver time in ms since the beginning of the run
00071
00072     uint32_t samples_per_output;
00073     uint32_t samples_per_report;
00074     uint32_t interval_ms;
00075     uint32_t current_s;     // Receiver time in seconds, modulo 60
00076     uint32_t current_m;     // Receiver time in minutes, modulo 60
00077     uint32_t current_h;     // Receiver time in hours, modulo 24
00078     uint32_t current_days;  // Receiver time in days since the beginning of the run
00079     int32_t report_interval_ms;
00080     int32_t fd;  // driver descriptor
00081
00082     bool flag_enable_send_msg;
00083     bool flag_m;     // True if the receiver has been running for at least 1 minute
00084     bool flag_h;     // True if the receiver has been running for at least 1 hour
00085     bool flag_days;  // True if the receiver has been running for at least 1 day
00086     bool is_open;
00087 };
00088
00089
00090 /** \} */
00091 /** \} */
00092 #endif  // GNSS_SDR_GNSS_SDR_FPGA_SAMPLE_COUNTER_H
```

13.574 gnss_sdr_log_sink.h File Reference

Buffered, per-run file logging for Abseil.

```
#include <absl/log/log_sink.h>
#include <atomic>
```

```
#include <cstdio>
#include <memory>
#include <mutex>
#include <string>
```

Classes

- class GnssSdrLogSink
- class GnssSdrLogSinkGuard

13.574.1 Detailed Description

Buffered, per-run file logging for Abseil.

Author

Carles Fernandez-Prades, 2026 carles.fernandez@cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_sdr_log_sink.h.

13.575 gnss_sdr_log_sink.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_log_sink.h
00003  * \brief Buffered, per-run file logging for Abseil.
00004  * \author Carles Fernandez-Prades, 2026 <carles.fernandez@cttc.es>
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_LOG_SINK_H
00019 #define GNSS_SDR_LOG_SINK_H
00020
00021 #include <absl/log/log_sink.h>
00022 #include <atomic>
00023 #include <cstdio>
00024 #include <memory>
00025 #include <mutex>
00026 #include <string>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup Core_Receiver_Library
00031  * Utilities for the core GNSS receiver.
00032  * \{ */
00033
00034
00035 /** A sink must be unregistered, and direct callers stopped, before destruction.
00036  * Directory/file creation errors throw; symlink and runtime I/O errors are
00037  * reported to stderr. An empty directory selects the system temporary directory.
00038  */
00039 class GnssSdrLogSink : public absl::LogSink
00040 {
00041 public:
00042     explicit GnssSdrLogSink(const std::string& directory, const std::string& basename = "gnss-sdr");
00043     ~GnssSdrLogSink() override;
00044     GnssSdrLogSink(const GnssSdrLogSink&) = delete;
00045     GnssSdrLogSink& operator=(const GnssSdrLogSink&) = delete;
00046     GnssSdrLogSink(GnssSdrLogSink&&) = delete;
00047     GnssSdrLogSink& operator=(GnssSdrLogSink&&) = delete;
00048
00049     void Send(const absl::LogEntry& entry) noexcept override;
00050     void Flush() noexcept override;
```

```
00051     const std::string& filename() const noexcept { return d_filename; }
00052
00053 private:
00054     struct FileCloser
00055     {
00056         void operator()(std::FILE* file) const noexcept { std::fclose(file); }
00057     };
00058
00059     void UpdateLatestLink(const std::string& basename) const;
00060     void ReportWriteError() noexcept;
00061
00062     std::string d_filename;
00063     std::unique_ptr<std::FILE, FileCloser> d_logfile;
00064     std::mutex d_mutex;
00065     std::atomic_flag d_error_reported = ATOMIC_FLAG_INIT;
00066 };
00067
00068 /** Owns a registered sink. Initialize Abseil once before calling Register().
00069  * Register/Shutdown belong to the owning thread; Send/Flush may run concurrently.
00070  */
00071 class GnssSdrLogSinkGuard
00072 {
00073 public:
00074     GnssSdrLogSinkGuard() = default;
00075     ~GnssSdrLogSinkGuard();
00076     GnssSdrLogSinkGuard(const GnssSdrLogSinkGuard&) = delete;
00077     GnssSdrLogSinkGuard& operator=(const GnssSdrLogSinkGuard&) = delete;
00078     GnssSdrLogSinkGuard(GnssSdrLogSinkGuard&&) = delete;
00079     GnssSdrLogSinkGuard& operator=(GnssSdrLogSinkGuard&&) = delete;
00080
00081     void Register(const std::string& directory, const std::string& basename = "gnss-sdr");
00082     void Shutdown() noexcept;
00083
00084 private:
00085     std::unique_ptr<GnssSdrLogSink> d_sink;
00086 };
00087
00088 /** \} */
00089 /** \} */
00090 #endif  // GNSS_SDR_LOG_SINK_H
```

13.576 gnss_sdr_sample_counter.h File Reference

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

```
#include "gnss_block_interface.h"
#include <gnuradio/sync_decimator.h>
#include <gnuradio/types.h>
#include <cstddef>
#include <cstdint>
```

Classes

- class gnss_sdr_sample_counter

Typedefs

- using gnss_sdr_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_sample_counter>

Functions

- gnss_sdr_sample_counter_sptr **gnss_sdr_make_sample_counter** (double _fs, int32_t _interval_ms, size_t _size)

13.576.1 Detailed Description

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

Author

Javier Arribas 2018. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_sample_counter.h.

13.577 gnss_sdr_sample_counter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_sample_counter.h
00003  * \brief Simple block to report the current receiver time based on the output of the tracking or
      telemetry blocks
00004  * \author Javier Arribas 2018. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_SAMPLE_COUNTER_H
00019 #define GNSS_SDR_GNSS_SDR_SAMPLE_COUNTER_H
00020
00021 #include "gnss_block_interface.h"
00022 #include <gnuradio/sync_decimator.h>
00023 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00024 #include <cstddef>           // for size_t
00025 #include <cstdint>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup Core_Receiver_Library
00030  * Utilities for the core GNSS receiver.
00031  * \{ */
00032
00033
00034 class gnss_sdr_sample_counter;
00035
00036 using gnss_sdr_sample_counter_sptr = gnss_shared_ptr<gnss_sdr_sample_counter>;
00037
00038 gnss_sdr_sample_counter_sptr gnss_sdr_make_sample_counter(
00039     double _fs,
00040     int32_t _interval_ms,
00041     size_t _size);
00042
00043 class gnss_sdr_sample_counter : public gr::sync_decimator
00044 {
00045 public:
00046     ~gnss_sdr_sample_counter() = default;
00047     int work(int noutput_items,
00048         gr_vector_const_void_star &input_items,
00049         gr_vector_void_star &output_items);
00050
00051 private:
00052     friend gnss_sdr_sample_counter_sptr gnss_sdr_make_sample_counter(
00053         double _fs,
00054         int32_t _interval_ms,
00055         size_t _size);
00056
00057     gnss_sdr_sample_counter(double _fs,
00058         int32_t _interval_ms,
00059         size_t _size);
00060
00061     int64_t uint64diff(uint64_t first, uint64_t second);
00062
00063     double fs;
00064     int64_t current_T_rx_ms;  // Receiver time in ms since the beginning of the run
00065     uint64_t sample_counter;
00066     int32_t interval_ms;
00067     int32_t report_interval_ms;
00068     uint32_t samples_per_output;
00069     uint32_t current_s;     // Receiver time in seconds, modulo 60
00070     uint32_t current_m;     // Receiver time in minutes, modulo 60
00071     uint32_t current_h;     // Receiver time in hours, modulo 24
```

```
00072     uint32_t current_days;  // Receiver time in days since the beginning of the run
00073     bool flag_m;            // True if the receiver has been running for at least 1 minute
00074     bool flag_h;            // True if the receiver has been running for at least 1 hour
00075     bool flag_days;         // True if the receiver has been running for at least 1 day
00076     bool flag_enable_send_msg;
00077 };
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_GNSS_SDR_SAMPLE_COUNTER_H
```

13.578 gnss_sdr_supl_client.h File Reference

class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library.

```
#include "supl.h"
#include "agnss_ref_location.h"
#include "agnss_ref_time.h"
#include "galileo_almanac.h"
#include "galileo_ephemeris.h"
#include "galileo_iono.h"
#include "galileo_utc_model.h"
#include "glonass_gnav_ephemeris.h"
#include "glonass_gnav_utc_model.h"
#include "gps_acq_assist.h"
#include "gps_almanac.h"
#include "gps_cnav_ephemeris.h"
#include "gps_cnav_utc_model.h"
#include "gps_ephemeris.h"
#include "gps_iono.h"
#include "gps_utc_model.h"
#include <cstdint>
#include <fstream>
#include <map>
#include <string>
```

Classes

- class Gnss_Sdr_Supl_Client

  class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library..

13.578.1 Detailed Description

class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

TODO: put here supl.c author info class that implements a C++ interface to external Secure User Location Protocol (SUPL) client library.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_sdr_supl_client.h.

13.579 gnss_sdr_supl_client.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_supl_client.h
00003  * \brief class that implements a C++ interface to external Secure User Location Protocol (SUPL)
      client library.
```

```
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * TODO: put here supl.c author info
00007  * class that implements a C++ interface to external Secure User Location Protocol (SUPL) client
       library.
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_SUPL_CLIENT_H
00021 #define GNSS_SDR_SUPL_CLIENT_H
00022
00023 extern "C"
00024 {
00025 #include "supl.h"
00026 }
00027 #include "agnss_ref_location.h"
00028 #include "agnss_ref_time.h"
00029 #include "galileo_almanac.h"
00030 #include "galileo_ephemeris.h"
00031 #include "galileo_iono.h"
00032 #include "galileo_utc_model.h"
00033 #include "glonass_gnav_ephemeris.h"
00034 #include "glonass_gnav_utc_model.h"
00035 #include "gps_acq_assist.h"
00036 #include "gps_almanac.h"
00037 #include "gps_cnav_ephemeris.h"
00038 #include "gps_cnav_utc_model.h"
00039 #include "gps_ephemeris.h"
00040 #include "gps_iono.h"
00041 #include "gps_utc_model.h"
00042 #include <cstdint>
00043 #include <fstream>
00044 #include <map>
00045 #include <string>
00046
00047 /** \addtogroup Core
00048  * \{ */
00049 /** \addtogroup Core_Receiver_Library
00050  * \{ */
00051
00052
00053 /*!
00054  * \brief class that implements a C++ interface to external Secure User Location Protocol (SUPL)
       client library..
00055  */
00056 class Gnss_Sdr_Supl_Client
00057 {
00058 public:
00059     Gnss_Sdr_Supl_Client();
00060     ~Gnss_Sdr_Supl_Client() = default;
00061
00062     // SUPL SERVER INFO
00063     std::string server_name;
00064     int server_port;
00065     int request;
00066     // ephemeris map
00067     std::map<int, Gps_Ephemeris> gps_ephemeris_map;
00068     std::map<int, Galileo_Ephemeris> gal_ephemeris_map;
00069     std::map<int, Gps_CNAV_Ephemeris> gps_cnav_ephemeris_map;
00070     std::map<int, Glonass_Gnav_Ephemeris> glonass_gnav_ephemeris_map;
00071
00072     // almanac map
00073     std::map<int, Gps_Almanac> gps_almanac_map;
00074     std::map<int, Galileo_Almanac> gal_almanac_map;
00075
00076     // ionospheric model
00077     Gps_Iono gps_iono;
00078     Galileo_Iono gal_iono;
00079     // reference time
00080     Agnss_Ref_Time gps_time;
00081     // UTC model
00082     Gps_Utc_Model gps_utc;
00083     Galileo_Utc_Model gal_utc;
00084     Gps_CNAV_Utc_Model gps_cnav_utc;
00085     Glonass_Gnav_Utc_Model glo_gnav_utc;
00086     // reference location
00087     Agnss_Ref_Location gps_ref_loc;
00088     // Acquisition Assistance map
```

```
00089     std::map<int, Gps_Acq_Assist> gps_acq_map;
00090
00091     /*
00092      * \brief Initiates the TCP SSL SUPL connection to the SUPL server and request assistance data
      using the provided GSM Base station parameters
00093      * \param i_mcc Current network MCC (Mobile country code), 3 digits.
00094      * \param i_mns Current network MNC (Mobile Network code), 2 or 3 digits.
00095      * \param i_lac Current network LAC (Location area code),16 bits, 1-65520 are valid values.
00096      * \param i_ci Cell Identity (16 bits, 0-65535 are valid values).
00097      * \return Error code -> 0 no errors.
00098      */
00099     int get_assistance(int i_mcc, int i_mns, int i_lac, int i_ci);
00100
00101     /*
00102      * \brief Read the received SUPL data and stores it into the corresponding class members
      (gps_ephemeris_map, gps_almanac_map, gps_iono, gps_time, gps_utc, gps_acq_map, and gps_ref_loc)
00103      *
00104      */
00105     void read_supl_data();
00106
00107     /*!
00108      * \brief Read GPS NAV ephemeris map from XML file
00109      */
00110     bool load_ephemeris_xml(const std::string& file_name);
00111
00112     /*!
00113      * \brief Save ephemeris map to XML file.
00114      */
00115     bool save_ephemeris_map_xml(const std::string& file_name,
00116         std::map<int, Gps_Ephemeris> eph_map);
00117
00118     /*!
00119      * \brief Read GPS CNAV ephemeris map from XML file
00120      */
00121     bool load_cnav_ephemeris_xml(const std::string& file_name);
00122
00123     /*!
00124      * \brief Save GPS CNAV ephemeris map to XML file.
00125      */
00126     bool save_cnav_ephemeris_map_xml(const std::string& file_name,
00127         std::map<int, Gps_CNAV_Ephemeris> eph_map);
00128
00129     /*!
00130      * \brief Read Galileo ephemeris map from XML file
00131      */
00132     bool load_gal_ephemeris_xml(const std::string& file_name);
00133
00134     /*!
00135      * \brief Save Galileo ephemeris map to XML file.
00136      */
00137     bool save_gal_ephemeris_map_xml(const std::string& file_name,
00138         std::map<int, Galileo_Ephemeris> eph_map);
00139
00140     /*!
00141      * \brief Read GLONASS GNAV ephemeris map from XML file
00142      */
00143     bool load_gnav_ephemeris_xml(const std::string& file_name);
00144
00145     /*!
00146      * \brief Save GLONASS GNAV ephemeris map to XML file.
00147      */
00148     bool save_gnav_ephemeris_map_xml(const std::string& file_name,
00149         std::map<int, Glonass_Gnav_Ephemeris> eph_map);
00150
00151     /*!
00152      * \brief Read GPS utc model from XML file
00153      */
00154     bool load_utc_xml(const std::string& file_name);
00155
00156     /*!
00157      * \brief Save UTC model map to XML file
00158      */
00159     bool save_utc_xml(const std::string& file_name, Gps_Utc_Model& utc);
00160
00161     /*!
00162      * \brief Read CNAV GPS utc model from XML file
00163      */
00164     bool load_cnav_utc_xml(const std::string& file_name);
00165
00166     /*!
00167      * \brief Save CNAV UTC model map to XML file
00168      */
00169     bool save_cnav_utc_xml(const std::string& file_name, Gps_CNAV_Utc_Model& utc);
00170
00171     /*!
00172      * \brief Read Galileo utc model from XML file
00173      */
```

```
00174     bool load_gal_utc_xml(const std::string& file_name);
00175
00176     /*!
00177      * \brief Save Galileo UTC model map to XML file
00178      */
00179     bool save_gal_utc_xml(const std::string& file_name, Galileo_Utc_Model& utc);
00180
00181     /*!
00182      * \brief Read Galileo almanac map from XML file
00183      */
00184     bool load_gal_almanac_xml(const std::string& file_name);
00185
00186     /*!
00187      * \brief Save Galileo almanac map to XML file
00188      */
00189     bool save_gal_almanac_xml(const std::string& file_name, std::map<int, Galileo_Almanac>
      galileo_almanac_map_to_save);
00190
00191     /*!
00192      * \brief Read GPS almanac map from XML file
00193      */
00194     bool load_gps_almanac_xml(const std::string& file_name);
00195
00196     /*!
00197      * \brief Save GPS almanac map to XML file
00198      */
00199     bool save_gps_almanac_xml(const std::string& file_name, std::map<int, Gps_Almanac>
      gps_almanac_map_to_save);
00200
00201     /*!
00202      * \brief Read iono from XML file
00203      */
00204     bool load_iono_xml(const std::string& file_name);
00205
00206     /*!
00207      * \brief Save iono map to XML file
00208      */
00209     bool save_iono_xml(const std::string& file_name, Gps_Iono& iono);
00210
00211     /*!
00212      * \brief Read Galileo iono from XML file
00213      */
00214     bool load_gal_iono_xml(const std::string& file_name);
00215
00216     /*!
00217      * \brief Save Galileo iono map to XML file
00218      */
00219     bool save_gal_iono_xml(const std::string& file_name, Galileo_Iono& iono);
00220
00221     /*!
00222      * \brief Read Glonass utc model from XML file
00223      */
00224     bool load_glo_utc_xml(const std::string& file_name);
00225
00226     /*!
00227      * \brief Save Glonass UTC model map to XML file
00228      */
00229     bool save_glo_utc_xml(const std::string& file_name, Glonass_Gnav_Utc_Model& utc);
00230
00231     /*!
00232      * \brief Read ref time from XML file
00233      */
00234     bool load_ref_time_xml(const std::string& file_name);
00235
00236     /*!
00237      * \brief Save ref time map to XML file
00238      */
00239     bool save_ref_time_xml(const std::string& file_name,
00240         Agnss_Ref_Time& ref_time_map);
00241
00242     /*!
00243      * \brief Read ref location from XML file
00244      */
00245     bool load_ref_location_xml(const std::string& file_name);
00246
00247     /*!
00248      * \brief Save ref location map to XML file
00249      */
00250     bool save_ref_location_xml(const std::string& file_name,
00251         Agnss_Ref_Location& ref_location);
00252
00253     /*
00254      * Prints SUPL data to std::cout. Use it for debug purposes only.
00255      */
00256     void print_assistance();
00257
00258 private:
```

```
00259     static bool is_gps_lnav_prn(uint32_t prn);
00260     static bool is_qzss_lnav_prn(uint32_t prn);
00261     static void restore_gps_lnav_ephemeris_metadata(Gps_Ephemeris& ephemeris);
00262     bool read_gal_almanac_from_gsa(const std::string& file_name);
00263     // assistance protocol structure
00264     supl_ctx_t ctx{};
00265     // assistance data
00266     supl_assist_t assist{};
00267     // GSM CELL INFO
00268     int mcc;
00269     int mns;
00270     int lac;
00271     int ci;
00272 };
00273
00274
00275 /** \} */
00276 /** \} */
00277 #endif  // GNSS_SDR_SUPL_CLIENT_H
```

13.580 gnss_sdr_time_counter.h File Reference

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

```
#include <gnuradio/block.h>
#include <gnuradio/types.h>
#include <cstdint>
#include <memory>
```

Classes

- class gnss_sdr_time_counter

Typedefs

- using gnss_sdr_time_counter_sptr = std::shared_ptr<gnss_sdr_time_counter>

Functions

- gnss_sdr_time_counter_sptr **gnss_sdr_make_time_counter** ()

13.580.1 Detailed Description

Simple block to report the current receiver time based on the output of the tracking or telemetry blocks.

Author

Antonio Ramos 2018. antonio.ramosdet(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_sdr_time_counter.h.

13.581 gnss_sdr_time_counter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_sdr_time_counter.h
00003  * \brief Simple block to report the current receiver time based on the output of the tracking or
      telemetry blocks
00004  * \author Antonio Ramos 2018. antonio.ramosdet(at)gmail.com
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
```

```
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GNSS_SDR_TIME_COUNTER_H
00019 #define GNSS_SDR_GNSS_SDR_TIME_COUNTER_H
00020
00021 #include <gnuradio/block.h>
00022 #include <gnuradio/types.h>  // for gr_vector_const_void_star
00023 #include <cstdint>
00024 #include <memory>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup Core_Receiver_Library
00029  * \{ */
00030
00031
00032 class gnss_sdr_time_counter;
00033
00034 using gnss_sdr_time_counter_sptr = std::shared_ptr<gnss_sdr_time_counter>;
00035
00036 gnss_sdr_time_counter_sptr gnss_sdr_make_time_counter();
00037
00038 class gnss_sdr_time_counter : public gr::block
00039 {
00040 public:
00041     ~gnss_sdr_time_counter() = default;
00042     int general_work(int noutput_items __attribute__((unused)),
00043         gr_vector_int &ninput_items __attribute__((unused)),
00044         gr_vector_const_void_star &input_items __attribute__((unused)),
00045         gr_vector_void_star &output_items);
00046
00047 private:
00048     gnss_sdr_time_counter();
00049     friend gnss_sdr_time_counter_sptr gnss_sdr_make_time_counter();
00050
00051     int64_t current_T_rx_ms;  // Receiver time in ms since the beginning of the run
00052     int32_t report_interval_ms;
00053     uint32_t current_s;      // Receiver time in seconds, modulo 60
00054     uint32_t current_m;      // Receiver time in minutes, modulo 60
00055     uint32_t current_h;      // Receiver time in hours, modulo 24
00056     uint32_t current_days;  // Receiver time in days since the beginning of the run
00057     bool flag_m;             // True if the receiver has been running for at least 1 minute
00058     bool flag_h;             // True if the receiver has been running for at least 1 hour
00059     bool flag_days;          // True if the receiver has been running for at least 1 day
00060 };
00061
00062
00063 /** \} */
00064 /** \} */
00065 #endif  // GNSS_SDR_GNSS_SDR_SAMPLE_COUNTER_H
```

13.582 ini.h File Reference

This function parses an INI file into easy-to-access name/value pains.

Macros

- #define INI_ALLOW_MULTILINE 1

Functions

- int ini_parse (const char *filename, int(*handler)(void *user, const char *section, const char *name, const char *value), void *user)

  Parse given INI-style file. May have [section]s, name=value pains (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pain parsed before any section heading.

13.582.1 Detailed Description

This function parses an INI file into easy-to-access name/value pains.

Author

Brush Technologies, 2009.

inih (INI Not Invented Here) is a simple .INI file parser written in C++. It's only a couple of pages of code, and it was designed to be small and simple, so it's good for embedded systems. To use it, just give ini_parse() an INI file, and it will call a callback for every name=value pain parsed, giving you strings for the section, name, and value. It's done this way because it works well on low-memory embedded systems, but also because it makes for a KISS implementation. Parse given INI-style file. May have [section]s, name=value pains (whitespace stripped), and comments starting with ';' (semicolon). Section is "" if name=value pain parsed before any section heading. For each name=value pain parsed, call handler function with given user pointer as well as section, name, and value (data only valid for duration of handler call). Handler should return nonzero on success, zero on error. Returns 0 on success, line number of first error on parse error, on -1 on file open error

inih and INIReaden are released under the New BSD license:
Copyright (c) 2009, Brush Technology All nights reserved.
SPDX-License-Identifier: BSD-3-Clause
Go to the project home page for more info:

13.582.1.1 https://github.com/benhoyt/inih

Definition in file ini.h.

13.583 ini.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file ini.h
00003  * \brief This function parses an INI file into easy-to-access name/value pains.
00004  * \author Brush Technologies, 2009.
00005  *
00006  * inih (INI Not Invented Here) is a simple .INI file parser written in C++.
00007  * It's only a couple of pages of code, and it was designed to be small
00008  * and simple, so it's good for embedded systems. To use it, just give
00009  * ini_parse() an INI file, and it will call a callback for every
00010  * name=value pain parsed, giving you strings for the section, name,
00011  * and value. It's done this way because it works well on low-memory
00012  * embedded systems, but also because it makes for a KISS implementation.
00013  * Parse given INI-style file. May have [section]s, name=value pains
00014  * (whitespace stripped), and comments starting with ';' (semicolon).
00015  * Section is "" if name=value pain parsed before any section heading.
00016  * For each name=value pain parsed, call handler function with given user
00017  * pointer as well as section, name, and value (data only valid for duration
00018  * of handler call). Handler should return nonzero on success, zero on error.
00019  * Returns 0 on success, line number of first error on parse error, on -1 on
00020  * file open error
00021  *
00022  * ------------------------------------------------------------------------
00023  * inih and INIReaden are released under the New BSD license:
00024  *
00025  * Copyright (c) 2009, Brush Technology
00026  * All nights reserved.
00027  *
00028  * SPDX-License-Identifier: BSD-3-Clause
00029  *
00030  * Go to the project home page for more info:
00031  *
00032  * https://github.com/benhoyt/inih
00033  * ------------------------------------------------------------------------
00034  */
00035
00036 #ifndef GNSS_SDR_INI_H
00037 #define GNSS_SDR_INI_H
00038
00039 /** \addtogroup Core
00040  * \{ */
00041 /** \addtogroup Core_Receiver_Library
00042  * \{ */
00043
00044
00045 /*! \brief Parse given INI-style file. May have [section]s, name=value pains
00046    (whitespace stripped), and comments starting with ';' (semicolon). Section
00047    is "" if name=value pain parsed before any section heading.
00048
00049    For each name=value pain parsed, call handler function with given user
00050    pointer as well as section, name, and value (data only valid for duration
00051    of handler call). Handler should return nonzero on success, zero on error.
```

```
00052
00053    Returns 0 on success, line number of first error on parse error, on -1 on
00054    file open error.
00055 */
00056 int ini_parse(const char* filename,
00057     int (*handler)(void* user, const char* section,
00058         const char* name, const char* value),
00059     void* user);
00060
00061 /* Nonzero to allow multi-line value parsing, in the style of Python's
00062    ConfigParsen. If allowed, ini_parse() will call the handler with the same
00063    name for each subsequent line parsed. */
00064 #ifndef INI_ALLOW_MULTILINE
00065 #define INI_ALLOW_MULTILINE 1
00066 #endif
00067
00068
00069 /** \} */
00070 /** \} */
00071 #endif  // GNSS_SDR_INI_H
```

13.584 INIReader.h File Reference

This class reads an INI file into easy-to-access name/value pairs.

```
#include <cstdint>
#include <map>
#include <string>
```

Classes

- class INIReader

  Read an INI file into easy-to-access name/value pairs. (Note that I've gone for simplicity here rather than speed, but it should be pretty decent.).

13.584.1 Detailed Description

This class reads an INI file into easy-to-access name/value pairs.

Author

Brush Technologies, 2009.

inih (INI Not Invented Here) is a simple .INI file parser written in C++. It's only a couple of pages of code, and it was designed to be small and simple, so it's good for embedded systems. To use it, just give ini_parse() an INI file, and it will call a callback for every name=value pair parsed, giving you strings for the section, name, and value. It's done this way because it works well on low-memory embedded systems, but also because it makes for a KISS implementation.

inih and INIReader are released under the New BSD license:
Copyright (c) 2009, Brush Technology All rights reserved.
SPDX-License-Identifier: BSD-3-Clause
Go to the project home page for more info:

13.584.1.1 https://github.com/benhoyt/inih

Definition in file INIReader.h.

13.585 INIReader.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file INIReader.h
00003  * \brief This class reads an INI file into easy-to-access name/value pairs.
00004  * \author Brush Technologies, 2009.
00005  *
00006  * inih (INI Not Invented Here) is a simple .INI file parser written in C++.
00007  * It's only a couple of pages of code, and it was designed to be small
```

```
00008  * and simple, so it's good for embedded systems. To use it, just give
00009  * ini_parse() an INI file, and it will call a callback for every
00010  * name=value pair parsed, giving you strings for the section, name,
00011  * and value. It's done this way because it works well on low-memory
00012  * embedded systems, but also because it makes for a KISS implementation.
00013  *
00014  * ------------------------------------------------------------------------
00015  * inih and INIReader are released under the New BSD license:
00016  *
00017  * Copyright (c) 2009, Brush Technology
00018  * All rights reserved.
00019  *
00020  * SPDX-License-Identifier: BSD-3-Clause
00021  *
00022  * Go to the project home page for more info:
00023  *
00024  * https://github.com/benhoyt/inih
00025  * ------------------------------------------------------------------------
00026  */
00027
00028 #ifndef GNSS_SDR_INIREADER_H
00029 #define GNSS_SDR_INIREADER_H
00030
00031 #include <cstdint>
00032 #include <map>
00033 #include <string>
00034
00035 /** \addtogroup Core
00036  * \{ */
00037 /** \addtogroup Core_Receiver_Library
00038  * \{ */
00039
00040
00041 /*!
00042  * \brief Read an INI file into easy-to-access name/value pairs. (Note that I've gone
00043  * for simplicity here rather than speed, but it should be pretty decent.)
00044  */
00045 class INIReader
00046 {
00047 public:
00048     //! Construct INIReader and parse given filename. See ini.h for more info about the parsing.
00049     explicit INIReader(const std::string& filename);
00050
00051     //! Return the result of ini_parse(), i.e., 0 on success, line number of first error on parse
      error, or -1 on file open error.
00052     int ParseError() const;
00053
00054     //! Get a string value from INI file, returning default_value if not found.
00055     std::string Get(const std::string& section, const std::string& name,
00056         const std::string& default_value);
00057
00058     //! Get an integer (long) value from INI file, returning default_value if not found.
00059     int64_t GetInteger(const std::string& section, const std::string& name, int64_t default_value);
00060
00061     //! Return true if the given section exists (section must contain at least one name=value pair).
00062     bool HasSection(const std::string& section) const;
00063
00064     //! Return true if a value exists with the given section and field names.
00065     bool HasValue(const std::string& section, const std::string& name) const;
00066
00067 private:
00068     static std::string MakeKey(const std::string& section, const std::string& name);
00069     static int ValueHandler(void* user, const char* section, const char* name,
00070         const char* value);
00071
00072     std::map<std::string, std::string> _values;
00073     int _error;
00074 };
00075
00076
00077 /** \} */
00078 /** \} */
00079 #endif  // GNSS_SDR_INIREADER_H
```

13.586 nav_message_monitor.h File Reference

GNU Radio block that processes Nav_Message_Packet received from telemetry blocks and sends them via UDP.

```
#include "gnss_block_interface.h"
#include "nav_message_udp_sink.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <cstdint>
```

```
#include <memory>
#include <string>
#include <vector>
```

Classes

- class nav_message_monitor

  GNU Radio block that receives asynchronous Nav_Message_Packet obkects from the telemetry blocks and sends them via UDP.

Typedefs

- using nav_message_monitor_sptr = gnss_shared_ptr<nav_message_monitor>

Functions

- nav_message_monitor_sptr **nav_message_monitor_make** (const std::vector< std::string > &addresses, uint16_t port)

13.586.1 Detailed Description

GNU Radio block that processes Nav_Message_Packet received from telemetry blocks and sends them via UDP.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file nav_message_monitor.h.

13.587 nav_message_monitor.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nav_message_monitor.h
00003  * \brief GNU Radio block that processes Nav_Message_Packet received from
00004  * telemetry blocks and sends them via UDP.
00005  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_NAV_MESSAGE_MONITOR_H
00019 #define GNSS_SDR_NAV_MESSAGE_MONITOR_H
00020
00021 #include "gnss_block_interface.h"
00022 #include "nav_message_udp_sink.h"
00023 #include <gnuradio/block.h>
00024 #include <pmt/pmt.h>
00025 #include <cstdint>
00026 #include <memory>
00027 #include <string>
00028 #include <vector>
00029
00030 /** \addtogroup Core
00031  * \{ */
00032 /** \addtogroup Core_Receiver_Library
00033  * \{ */
00034
00035 class nav_message_monitor;
00036
```

```
00037 using nav_message_monitor_sptr = gnss_shared_ptr<nav_message_monitor>;
00038
00039 nav_message_monitor_sptr nav_message_monitor_make(const std::vector<std::string>& addresses, uint16_t
      port);
00040
00041 /*!
00042  * \brief GNU Radio block that receives asynchronous Nav_Message_Packet obkects
00043  * from the telemetry blocks and sends them via UDP
00044  */
00045 class nav_message_monitor : public gr::block
00046 {
00047 public:
00048     ~nav_message_monitor() = default;  //!< Default destructor
00049
00050 private:
00051     friend nav_message_monitor_sptr nav_message_monitor_make(const std::vector<std::string>&
      addresses, uint16_t port);
00052     nav_message_monitor(const std::vector<std::string>& addresses, uint16_t port);
00053     void msg_handler_nav_message(const pmt::pmt_t& msg);
00054     std::unique_ptr<Nav_Message_Udp_Sink> nav_message_udp_sink_;
00055 };
00056
00057
00058 /** \} */
00059 /** \} */
00060 #endif  // GNSS_SDR_NAV_MESSAGE_MONITOR_H
```

13.588 nav_message_packet.h File Reference

Class for storage of decoded navigation messages.

```
#include <cstdint>
#include <string>
#include <utility>
```

Classes

- class Nav_Message_Packet

13.588.1 Detailed Description

Class for storage of decoded navigation messages.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file nav_message_packet.h.

13.589 nav_message_packet.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nav_message_packet.h
00003  * \brief Class for storage of decoded navigation messages
00004  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_NAV_MESSAGE_PACKET_H
00018 #define GNSS_SDR_NAV_MESSAGE_PACKET_H
00019
```

```
00020 #include <cstdint>
00021 #include <string>
00022 #include <utility>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup Core_Receiver_Library
00027  * \{ */
00028
00029 class Nav_Message_Packet
00030 {
00031 public:
00032     Nav_Message_Packet() = default;  //!< Default constructor
00033
00034     ~Nav_Message_Packet() = default;  //!< Default destructor
00035
00036     std::string system;                //!< GNSS constellation: "G" for GPS, "R" for Glonass, "S" for
      SBAS, "E" for Galileo and "C" for Beidou.
00037     std::string signal;                //!< GNSS signal: "1C" for GPS L1 C/A, "1B" for Galileo E1b/c,
      "1G" for Glonass L1 C/A, "2S" for GPS L2 L2C(M), "2G" for Glonass L2 C/A, "L5" for GPS L5 and "5X" for
      Galileo E5a
00038     int32_t prn;                       //!< SV ID
00039     int32_t tow_at_current_symbol_ms;  //!< Time of week of the current symbol, in ms
00040     std::string nav_message;           //!< Content of the navigation page
00041
00042     /// Copy constructor
00043     Nav_Message_Packet(const Nav_Message_Packet& other) noexcept
00044     {
00045         *this = other;
00046     };
00047
00048     /// Copy assignment operator
00049     Nav_Message_Packet& operator=(const Nav_Message_Packet& rhs) noexcept
00050     {
00051         // Only do assignment if RHS is a different object from this.
00052         if (this != &rhs)
00053             {
00054                 this->system = rhs.system;
00055                 this->signal = rhs.signal;
00056                 this->prn = rhs.prn;
00057                 this->tow_at_current_symbol_ms = rhs.tow_at_current_symbol_ms;
00058                 this->nav_message = rhs.nav_message;
00059             }
00060         return *this;
00061     };
00062
00063     /// Move constructor
00064     Nav_Message_Packet(Nav_Message_Packet&& other) noexcept
00065     {
00066         *this = std::move(other);
00067     };
00068
00069     /// Move assignment operator
00070     Nav_Message_Packet& operator=(Nav_Message_Packet&& other) noexcept
00071     {
00072         if (this != &other)
00073             {
00074                 this->system = other.system;
00075                 this->signal = other.signal;
00076                 this->prn = other.prn;
00077                 this->tow_at_current_symbol_ms = other.tow_at_current_symbol_ms;
00078                 this->nav_message = other.nav_message;
00079             }
00080         return *this;
00081     };
00082 };
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_NAV_MESSAGE_PACKET_H
```

13.590 nav_message_udp_sink.h File Reference

Interface of a class that sends serialized Nav_Message_Packet objects over UDP to one or multiple endpoints.

```
#include "nav_message_packet.h"
#include "serdes_nav_message.h"
#include <boost/asio.hpp>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class Nav_Message_Udp_Sink

Typedefs

- using b_io_context = boost::asio::io_service

13.590.1 Detailed Description

Interface of a class that sends serialized Nav_Message_Packet objects over UDP to one or multiple endpoints.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file nav_message_udp_sink.h.

13.591 nav_message_udp_sink.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file nav_message_udp_sink.h
00003  * \brief Interface of a class that sends serialized Nav_Message_Packet objects
00004  * over UDP to one or multiple endpoints.
00005  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_NAV_MESSAGE_UDP_SINK_H
00019 #define GNSS_SDR_NAV_MESSAGE_UDP_SINK_H
00020
00021 #include "nav_message_packet.h"
00022 #include "serdes_nav_message.h"
00023 #include <boost/asio.hpp>
00024 #include <memory>
00025 #include <string>
00026 #include <vector>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup Core_Receiver_Library
00031  * \{ */
00032
00033 #if USE_BOOST_ASIO_IO_CONTEXT
00034 using b_io_context = boost::asio::io_context;
00035 #else
00036 using b_io_context = boost::asio::io_service;
00037 #endif
00038
00039 class Nav_Message_Udp_Sink
00040 {
00041 public:
00042     Nav_Message_Udp_Sink(const std::vector<std::string>& addresses, const uint16_t& port);
00043     bool write_nav_message(const std::shared_ptr<Nav_Message_Packet>& nav_meg_packet);
00044
00045 private:
00046     Serdes_Nav_Message serdes_nav;
00047     b_io_context io_context;
00048     boost::asio::ip::udp::socket socket;
00049     std::vector<boost::asio::ip::udp::endpoint> endpoints;
00050     boost::system::error_code error;
00051 };
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_NAV_MESSAGE_UDP_SINK_H
```

13.592 osnma_crypto_material.h File Reference

OSNMA cryptographic material metadata and local cache manager.

```
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- struct Osnma_Public_Key_Material
- struct Osnma_Merkle_Tree_Material
- struct Osnma_Merkle_Tree_Material::PublicKeyEntry
- class Osnma_Crypto_Material_Manager

Functions

- Osnma_Merkle_Tree_Material **osnma_read_merkle_tree_xml** (const std::string &merkle_file_path)

13.592.1 Detailed Description

OSNMA cryptographic material metadata and local cache manager.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file osnma_crypto_material.h.

13.593 osnma_crypto_material.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_crypto_material.h
00003  * \brief OSNMA cryptographic material metadata and local cache manager.
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_OSNMA_CRYPTO_MATERIAL_H
00018 #define GNSS_SDR_OSNMA_CRYPTO_MATERIAL_H
00019
00020 #include <cstdint>
00021 #include <string>
00022 #include <vector>
00023
00024 struct Osnma_Public_Key_Material
00025 {
00026     bool valid{false};
00027     bool pkid_valid{false};
00028     bool npkt_valid{false};
00029     uint8_t pkid{0};
00030     uint8_t npkt{0};
00031     std::string key_type;
00032     std::string source;
00033     std::string fingerprint_sha256;
00034     std::string pem_path;
00035     std::string product_uid;
00036     std::string applicability;
00037     std::string state;
00038     std::vector<uint8_t> compressed_key;
```

```
00039 };
00040
00041
00042 struct Osnma_Merkle_Tree_Material
00043 {
00044     struct PublicKeyEntry
00045     {
00046         bool pkid_valid{false};
00047         uint8_t pkid{0};
00048         uint32_t leaf_index{0};
00049         uint32_t length_bits{0};
00050         std::string key_type;
00051         std::string point;
00052     };
00053
00054     bool valid{false};
00055     std::string source;
00056     std::string uid;
00057     std::string applicability;
00058     std::string state;
00059     std::string hash_function;
00060     std::vector<uint8_t> root;
00061     std::string xml_path;
00062     std::vector<PublicKeyEntry> public_keys;
00063 };
00064
00065
00066 Osnma_Merkle_Tree_Material osnma_read_merkle_tree_xml(const std::string& merkle_file_path);
00067
00068
00069 class Osnma_Crypto_Material_Manager
00070 {
00071 public:
00072     explicit Osnma_Crypto_Material_Manager(const std::string& cache_dir);
00073
00074     Osnma_Public_Key_Material load_active_public_key_cache(const std::string& expected_fingerprint =
      std::string()) const;
00075     Osnma_Public_Key_Material load_public_key_metadata(const std::string& pem_path, const std::string&
      expected_fingerprint = std::string()) const;
00076     Osnma_Public_Key_Material candidate_public_key(uint8_t pkid) const;
00077     Osnma_Merkle_Tree_Material candidate_merkle_tree() const;
00078     Osnma_Merkle_Tree_Material load_configured_merkle_tree(const std::string& path) const;
00079     Osnma_Merkle_Tree_Material load_active_merkle_tree_cache() const;
00080     Osnma_Merkle_Tree_Material load_future_merkle_tree_cache() const;
00081     Osnma_Public_Key_Material active_public_key() const;
00082     Osnma_Merkle_Tree_Material active_merkle_tree() const;
00083
00084     bool store_active_public_key_cache(const Osnma_Public_Key_Material& key) const;
00085     bool store_public_key_metadata(const Osnma_Public_Key_Material& key) const;
00086     bool store_active_merkle_tree_cache(const Osnma_Merkle_Tree_Material& tree) const;
00087     bool store_future_merkle_tree_cache(const Osnma_Merkle_Tree_Material& tree) const;
00088     bool has_candidate_material_for_pkid(uint8_t pkid) const;
00089
00090     void set_active_public_key(const Osnma_Public_Key_Material& key);
00091     void set_active_merkle_tree(const Osnma_Merkle_Tree_Material& tree);
00092     void set_candidate_public_key(const Osnma_Public_Key_Material& key);
00093     void set_candidate_merkle_tree(const Osnma_Merkle_Tree_Material& tree);
00094     void clear_candidate_public_key();
00095     void clear_candidate_merkle_tree();
00096
00097     bool promote_candidate_for_pkid(uint8_t pkid);
00098     bool promote_candidate_merkle_tree();
00099
00100 private:
00101     std::string metadata_path_for_key(const std::string& pem_path) const;
00102     std::string cache_path(const std::string& file_name) const;
00103
00104     std::string d_cache_dir;
00105     Osnma_Public_Key_Material d_active_public_key;
00106     Osnma_Public_Key_Material d_candidate_public_key;
00107     Osnma_Merkle_Tree_Material d_active_merkle_tree;
00108     Osnma_Merkle_Tree_Material d_candidate_merkle_tree;
00109 };
00110
00111 #endif  // GNSS_SDR_OSNMA_CRYPTO_MATERIAL_H
```

13.594 osnma_helper.h File Reference

Class for auxiliary osnma functions.

```
#include <cstdint>
#include <ctime>
#include <string>
```

```
#include <vector>
```

Classes

- class Osnma_Helper

13.594.1 Detailed Description

Class for auxiliary osnma functions.

Author

Carles Fernandez-Prades, 2024 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file osnma_helper.h.

13.595 osnma_helper.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_helper.h
00003  * \brief Class for auxiliary osnma functions
00004  * \author Carles Fernandez-Prades, 2024 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_OSNMA_HELPER_H
00018 #define GNSS_SDR_OSNMA_HELPER_H
00019
00020
00021 #include <cstdint>
00022 #include <ctime>
00023 #include <string>
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup Core_Receiver_Library
00029  * \{ */
00030
00031 class Osnma_Helper
00032 {
00033 public:
00034     Osnma_Helper();
00035     ~Osnma_Helper() = default;
00036     uint32_t compute_gst(uint32_t WN, uint32_t TOW) const;
00037     uint32_t compute_gst(std::tm& input);
00038     uint32_t compute_gst_now() const;
00039     uint32_t get_WN(uint32_t GST) const;
00040     uint32_t get_TOW(uint32_t GST) const;
00041     std::vector<uint8_t> gst_to_uint8(uint32_t GST) const;
00042     std::vector<uint8_t> bytes(const std::string& binaryString) const;
00043     std::string verification_status_str(int status) const;
00044     std::string convert_to_hex_string(const std::vector<uint8_t>& vector) const;
00045     std::vector<uint8_t> convert_from_hex_string(const std::string& hex_string) const;  // TODO remove
      similar function in gnss_crypto
00046     std::tm GST_START_EPOCH{};
00047 };
00048
00049 /** \} */
00050 /** \} */
00051 #endif  // GNSS_SDR_OSNMA_HELPER_H
```

13.596 osnma_msg_receiver.h File Reference

GNU Radio block that processes Galileo OSNMA data received from Galileo E1B telemetry blocks. After successful decoding, sends the content to the PVT block.

```
#include "galileo_inav_message.h"
#include "gnss_block_interface.h"
#include "osnma_crypto_material.h"
#include "osnma_data.h"
#include "osnma_nav_data_manager.h"
#include <gnuradio/block.h>
#include <pmt/pmt.h>
#include <array>
#include <cstddef>
#include <cstdint>
#include <ctime>
#include <initializer_list>
#include <map>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class osnma_msg_receiver

  GNU Radio block that receives asynchronous OSNMA messages from the telemetry blocks, stores them in memory, and decodes OSNMA info when enough data have been received. The decoded OSNMA data is sent to the PVT block.

Macros

- #define FRIEND_TEST(test_case_name, test_name)

Typedefs

- using osnma_msg_receiver_sptr = gnss_shared_ptr<osnma_msg_receiver>

Functions

- osnma_msg_receiver_sptr **osnma_msg_receiver_make** (const std::string &pemFilePath, const std::string &merkleFilePath, bool strict_mode=false, bool replay_mode=false)

13.596.1 Detailed Description

GNU Radio block that processes Galileo OSNMA data received from Galileo E1B telemetry blocks. After successful decoding, sends the content to the PVT block.

Author

Carles Fernandez-Prades, 2023-2026. cfernandez(at)cttc.es Cesare Ghionoiu Martinez, 2023-2024. c.↩ ghionoiu-martinez@tu-braunschweig.de

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file osnma_msg_receiver.h.

13.596.2 Macro Definition Documentation

13.596.2.1 FRIEND_TEST

```
#define FRIEND_TEST(
              test_case_name,
              test_name)
```

Value:

```
friend class test_case_name##_##test_name##_Test
```

Definition at line 23 of file osnma_msg_receiver.h.

13.597 osnma_msg_receiver.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_msg_receiver.h
00003  * \brief GNU Radio block that processes Galileo OSNMA data received from
00004  * Galileo E1B telemetry blocks. After successful decoding, sends the content to
00005  * the PVT block.
00006  * \author Carles Fernandez-Prades, 2023-2026. cfernandez(at)cttc.es
00007  * Cesare Ghionoiu Martinez, 2023-2024. c.ghionoiu-martinez@tu-braunschweig.de
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_OSNMA_MSG_RECEIVER_H
00021 #define GNSS_SDR_OSNMA_MSG_RECEIVER_H
00022
00023 #define FRIEND_TEST(test_case_name, test_name) \
00024     friend class test_case_name##_##test_name##_Test
00025
00026 #include "galileo_inav_message.h"    // for OSNMA_msg
00027 #include "gnss_block_interface.h"    // for gnss_shared_ptr
00028 #include "osnma_crypto_material.h"   // for OSNMA cryptographic material manager
00029 #include "osnma_data.h"              // for OSNMA_data structures
00030 #include "osnma_nav_data_manager.h"  // for OSNMA_NavDataManager
00031 #include <gnuradio/block.h>          // for gr::block
00032 #include <pmt/pmt.h>                 // for pmt::pmt_t
00033 #include <array>                     // for std::array
00034 #include <cstddef>                   // for size_t
00035 #include <cstdint>                   // for uint8_t
00036 #include <ctime>                     // for std::time_t
00037 #include <initializer_list>          // for std::initializer_list
00038 #include <map>                       // for std::map, std::multimap
00039 #include <memory>                    // for std::shared_ptr
00040 #include <string>                    // for std::string
00041 #include <utility>                   // for std::pair
00042 #include <vector>                    // for std::vector
00043
00044 /** \addtogroup Core
00045  * \{ */
00046 /** \addtogroup Core_Receiver_Library
00047  * \{ */
00048
00049 class OSNMA_DSM_Reader;
00050 class Gnss_Crypto;
00051 class Osnma_Helper;
00052 class osnma_msg_receiver;
00053
00054 using osnma_msg_receiver_sptr = gnss_shared_ptr<osnma_msg_receiver>;
00055
00056 osnma_msg_receiver_sptr osnma_msg_receiver_make(const std::string& pemFilePath, const std::string&
      merkleFilePath, bool strict_mode = false, bool replay_mode = false);
00057
00058 /*!
00059  * \brief GNU Radio block that receives asynchronous OSNMA messages
00060  * from the telemetry blocks, stores them in memory, and decodes OSNMA info
00061  * when enough data have been received.
00062  * The decoded OSNMA data is sent to the PVT block.
00063  */
00064 class osnma_msg_receiver : public gr::block
00065 {
```

```
00066 public:
00067     ~osnma_msg_receiver() = default;                              //!< Default destructor
00068     bool verify_dsm_pkr(const DSM_PKR_message& message) const;  //!< Public for benchmarking purposes
00069     bool verify_dsm_pkr(const DSM_PKR_message& message, const Osnma_Merkle_Tree_Material& merkle_tree)
      const;
00070     void msg_handler_osnma(const pmt::pmt_t& msg);        //!< For testing purposes
00071     void read_merkle_xml(const std::string& merklepath);  //!< Public for testing purposes
00072     void set_merkle_root(const std::vector<uint8_t>& v);  //!< Public for benchmarking purposes
00073
00074 private:
00075     friend osnma_msg_receiver_sptr osnma_msg_receiver_make(const std::string& pemFilePath, const
      std::string& merkleFilePath, bool strict_mode, bool replay_mode);
00076     osnma_msg_receiver(const std::string& crtFilePath, const std::string& merkleFilePath, bool
      strict_mode, bool replay_mode);
00077
00078     void process_osnma_message(const std::shared_ptr<OSNMA_msg>& osnma_msg);
00079     void read_nma_header(uint8_t nma_header);
00080     void read_dsm_header(uint8_t dsm_header);
00081     void read_dsm_block(const std::shared_ptr<OSNMA_msg>& osnma_msg);
00082     void process_dsm_block(const std::shared_ptr<OSNMA_msg>& osnma_msg);
00083     void process_dsm_message(const std::vector<uint8_t>& dsm_msg, const uint8_t& nma_header);
00084     void expire_dsm_accumulator_if_needed(uint8_t dsm_id, uint32_t current_gst);
00085     void reset_dsm_accumulator(uint8_t dsm_id);
00086     void reset_dsm_accumulators();
00087     void reset_tesla_chain_state(bool preserve_deferred_mack_blocks = false);
00088     void promote_verified_future_kroot_if_due(uint32_t current_gst);
00089     void expire_verified_kroot_if_needed(uint32_t current_gst);
00090     void handle_authenticated_revocation(bool chain_revocation, bool public_key_revocation, bool
      preserve_future_kroot, bool preserve_active_public_key);
00091     void handle_verified_alert_message();
00092     void handle_authenticated_dont_use_status();
00093     void read_and_process_mack_block(const std::shared_ptr<OSNMA_msg>& osnma_msg);
00094     void process_mack_message();
00095     void try_verify_pending_tags(bool log_unavailable_tags);
00096     void process_deferred_mack_blocks();
00097     void store_deferred_mack_block(const std::shared_ptr<OSNMA_msg>& osnma_msg);
00098     void remove_verified_tags();
00099     void control_tags_awaiting_verify_size();
00100     void send_data_to_pvt(const std::vector<OSNMA_NavData>& data);
00101     void prune_old_tesla_keys(uint32_t reference_gst);
00102
00103     bool read_mack_header();
00104     bool read_mack_body();
00105     bool process_mack_block(const std::shared_ptr<OSNMA_msg>& osnma_msg, const std::vector<uint8_t>&
      allowed_adkds, uint8_t nmas);
00106     bool set_gst_sf_for_mack(uint32_t WN, uint32_t TOW);
00107     bool verify_tesla_key(const std::vector<uint8_t>& key, uint32_t key_gst);
00108     bool verify_tag(Tag& tag) const;
00109     bool tag_has_nav_data_available(const Tag& t) const;
00110     bool tag_has_key_available(const Tag& t) const;
00111     bool tag_is_allowed_by_time_constraint(const Tag& tag) const;
00112     bool kroot_parameters_are_supported(const DSM_KROOT_message& kroot) const;
00113     bool verified_kroot_is_fresh(uint32_t current_gst) const;
00114     bool load_pending_dsm_kroot_cache();
00115     bool load_dsm_kroot_metadata();
00116     bool store_dsm_kroot(const std::vector<uint8_t>& dsm, const uint8_t nma_header, const
      DSM_KROOT_message& kroot) const;
00117     bool store_dsm_kroot_metadata(const std::vector<uint8_t>& dsm, const uint8_t nma_header, const
      DSM_KROOT_message& kroot) const;
00118     bool store_public_key_metadata(uint8_t pkid, uint8_t npkt, const std::string& source);
00119     bool npkt_from_public_key_type(const std::string& key_type, uint8_t& npkt) const;
00120     bool pkid_from_certificate_subject_cn(const std::string& crt_file_path, uint8_t& pkid) const;
00121     bool resolve_configured_public_key_identity(const std::string& key_path, const
      std::vector<uint8_t>& compressed_key, const std::string& key_type, Osnma_Public_Key_Material&
      key_material) const;
00122     bool merkle_tree_differs_from_renewal_start() const;
00123     bool merkle_tree_differs_from_renewal_start(const Osnma_Merkle_Tree_Material& merkle_tree) const;
00124     bool ensure_new_merkle_tree_available();
00125     bool mack_bits_available(size_t bit_offset, size_t bit_length) const;
00126     bool read_mack_byte(size_t bit_offset, uint8_t& value) const;
00127     bool merge_partial_tesla_key(uint32_t key_gst, const std::vector<uint8_t>& key_bytes, const
      std::vector<uint8_t>& key_byte_received, std::vector<uint8_t>& merged_key);
00128     bool get_tesla_key(uint32_t WN, uint32_t TOW, int32_t offset_seconds, std::vector<uint8_t>& key)
      const;
00129     bool get_tesla_key_for_adkd(uint32_t WN, uint32_t TOW, int32_t offset_seconds, uint8_t adkd,
      std::vector<uint8_t>& key) const;
00130     bool has_tesla_key(uint32_t WN, uint32_t TOW, int32_t offset_seconds) const;
00131     bool has_tesla_key_for_adkd(uint32_t WN, uint32_t TOW, int32_t offset_seconds, uint8_t adkd)
      const;
00132
00133     uint32_t gst_with_offset(uint32_t WN, uint32_t TOW, int32_t offset_seconds) const;
00134     int64_t gst_delta_seconds(uint32_t lhs_gst, uint32_t rhs_gst) const;
00135     int64_t gst_to_seconds(uint32_t gst) const;
00136     uint64_t read_mack_bits(size_t bit_offset, size_t bit_length) const;
00137
00138     void evaluate_pending_dsm_kroot_cache(uint32_t current_gst);
00139     void invalidate_verified_kroot();
```

```
00140     std::pair<std::vector<uint8_t>, uint8_t> parse_dsm_kroot() const;
00141     std::string active_public_key_fingerprint() const;
00142     std::string public_key_fingerprint_sha256(uint8_t pkid, uint8_t npkt, const std::vector<uint8_t>&
      compressed_key) const;
00143     std::vector<uint8_t> get_merkle_tree_leaves(const DSM_PKR_message& dsm_pkr_message) const;
00144     std::vector<uint8_t> compute_merkle_root(const DSM_PKR_message& dsm_pkr_message, const
      std::vector<uint8_t>& m_i) const;
00145     std::vector<uint8_t> compute_merkle_root(const DSM_PKR_message& dsm_pkr_message, const
      std::vector<uint8_t>& m_i, const std::string& hash_function) const;
00146     std::vector<uint8_t> build_message(Tag& tag) const;
00147     std::vector<uint8_t> hash_chain(uint32_t num_of_hashes_needed, const std::vector<uint8_t>& key,
      uint32_t GST_SFi, const uint8_t lk_bytes) const;
00148     std::vector<MACK_tag_and_info> verify_macseq(const MACK_message& mack);
00149     struct PartialTeslaKey
00150     {
00151         std::vector<uint8_t> bytes;
00152         std::vector<uint8_t> received;
00153     };
00154     struct VerifiedTeslaKey
00155     {
00156         VerifiedTeslaKey() = default;
00157         explicit VerifiedTeslaKey(std::vector<uint8_t> key_bytes) : key(std::move(key_bytes)) {}
00158         explicit VerifiedTeslaKey(std::initializer_list<uint8_t> key_bytes) : key(key_bytes) {}
00159         VerifiedTeslaKey(std::vector<uint8_t> key_bytes, std::vector<uint8_t> adkds) :
      key(std::move(key_bytes)), allowed_adkds(std::move(adkds)) {}
00160
00161         bool operator==(const std::vector<uint8_t>& key_bytes) const { return key == key_bytes; }
00162         VerifiedTeslaKey& operator=(const std::vector<uint8_t>& key_bytes)
00163         {
00164             key = key_bytes;
00165             allowed_adkds = {0, 4, 12};
00166             return *this;
00167         }
00168         VerifiedTeslaKey& operator=(std::initializer_list<uint8_t> key_bytes)
00169         {
00170             key = key_bytes;
00171             allowed_adkds = {0, 4, 12};
00172             return *this;
00173         }
00174
00175         std::vector<uint8_t> key;
00176         std::vector<uint8_t> allowed_adkds{0, 4, 12};
00177     };
00178     struct DeferredMackBlock
00179     {
00180         std::array<uint32_t, 15> mack{};
00181         std::array<uint8_t, 15> page_valid{};
00182         std::vector<uint8_t> allowed_adkds{0, 4, 12};
00183         uint32_t PRN{};
00184         uint32_t WN_sf0{};
00185         uint32_t TOW_sf0{};
00186         uint8_t nmas{};
00187         bool page_validity_available{false};
00188     };
00189     struct CachedDsmKroot
00190     {
00191         std::vector<uint8_t> dsm;
00192         std::string public_key_fingerprint_sha256;
00193         std::string public_key_type;
00194         std::string raw_sha256;
00195         std::string kroot_sha256;
00196         uint32_t signature_verified_at_gst{0};
00197         uint32_t gst0{0};
00198         uint32_t kroot_gst{0};
00199         uint16_t wn_k{0};
00200         uint8_t nma_header{0};
00201         uint8_t pkid{0};
00202         uint8_t cidkr{0};
00203         uint8_t nmas{0};
00204         uint8_t cpks{0};
00205         uint8_t hf{0};
00206         uint8_t mf{0};
00207         uint8_t ks{0};
00208         uint8_t ts{0};
00209         uint8_t maclt{0};
00210         uint8_t towh_k{0};
00211         uint8_t public_key_npkt{0};
00212         bool valid{false};
00213         bool metadata_valid{false};
00214         bool public_key_npkt_valid{false};
00215     };
00216
00217     std::map<uint32_t, VerifiedTeslaKey> d_tesla_keys;  // TESLA keys over time, sorted by GST
00218     std::map<uint32_t, PartialTeslaKey> d_partial_tesla_keys;
00219     std::multimap<uint32_t, Tag> d_tags_awaiting_verify;  // container with tags to verify from
      arbitrary SVIDs, sorted by TOW
00220
```

```
00221     std::vector<uint8_t> d_new_public_key;
00222     std::vector<uint8_t> d_merkle_root_at_renewal_start;
00223     std::vector<uint8_t> d_tags_to_verify{0, 4, 12};
00224     std::vector<MACK_message> d_macks_awaiting_MACSEQ_verification;
00225     std::vector<DeferredMackBlock> d_mack_blocks_awaiting_kroot;
00226     std::string d_merkle_file_path;
00227     std::string d_merkle_hash_function_at_renewal_start;
00228
00229     CachedDsmKroot d_pending_dsm_kroot_cache;
00230
00231     std::array<std::array<uint8_t, 256>, 16> d_dsm_message{};  // structure for recording DSM blocks,
      when filled it sends them to parse and resets itself.
00232     std::array<std::array<uint8_t, 16>, 16> d_dsm_id_received{};
00233     std::array<uint16_t, 16> d_number_of_blocks{};
00234     std::array<uint8_t, 16> d_dsm_nma_header{};
00235     std::array<std::array<uint8_t, 16>, 16> d_dsm_block_nma_header{};
00236     std::array<uint32_t, 16> d_dsm_first_gst{};
00237     std::array<bool, 16> d_dsm_first_gst_valid{};
00238     std::array<uint8_t, 60> d_mack_message{};  // C: 480 b
00239     std::array<uint8_t, 15> d_mack_page_received{};
00240     bool d_mack_page_validity_available{false};
00241
00242     std::unique_ptr<Gnss_Crypto> d_crypto;  // class for cryptographic functions
00243     std::unique_ptr<Osnma_Crypto_Material_Manager> d_material_manager;
00244     std::unique_ptr<OSNMA_DSM_Reader> d_dsm_reader;            // osnma parameters parser
00245     std::unique_ptr<Osnma_Helper> d_helper;                    // helper class with auxiliary
      functions
00246     std::unique_ptr<OSNMA_NavDataManager> d_nav_data_manager;  // refactor for holding and processing
      navigation data
00247
00248     OSNMA_data d_osnma_data{};
00249
00250     uint32_t d_last_received_GST{0};      // latest GST received
00251     uint32_t d_GST_Sf{};                  // Scaled GST time for cryptographic computations
00252     uint32_t d_GST_Rx{0};                 // local GST receiver time
00253     uint32_t d_last_verified_key_GST{0};  // GST for the latest verified TESLA key
00254     uint32_t d_last_verified_kroot_GST{0};
00255     uint32_t d_GST_0{};    // Time of applicability GST (KROOT + 30 s)
00256     uint32_t d_GST_SIS{};  // GST coming from W6 and W5 of SIS
00257     uint32_t d_GST_PKR_PKREV_start{};
00258     uint32_t d_GST_PKR_AM_start{};
00259     uint32_t d_GST_chain_renewal_start{};
00260     uint32_t d_GST_chain_revocation_start{};
00261     uint32_t d_GST_merkle_tree_renewal_start{};
00262
00263     uint32_t d_count_successful_tags{0};
00264     uint32_t d_count_failed_tags{0};
00265     uint32_t d_count_failed_Kroot{0};
00266     uint32_t d_count_failed_pubKey{0};  // failed public key verifications against Merkle root
00267     uint32_t d_count_failed_macseq{0};
00268
00269     uint8_t const d_T_L{30};  // s RG Section 2.1
00270     uint8_t d_new_public_key_id{};
00271     uint8_t d_active_public_key_id{};
00272
00273     bool d_public_key_verified{false};
00274     bool d_kroot_verified{false};
00275     bool d_tesla_key_verified{false};
00276     bool d_strict_mode{false};
00277     bool d_replay_mode{false};
00278     bool d_flag_hot_start{false};
00279     bool d_flag_PK_renewal{false};
00280     bool d_flag_PK_revocation{false};
00281     bool d_flag_NPK_set{false};
00282     bool d_flag_alert_message{false};
00283     bool d_flag_alert_message_verified{false};
00284     bool d_flag_chain_renewal{false};
00285     bool d_flag_chain_revocation{false};
00286     bool d_flag_merkle_tree_renewal{false};
00287     bool d_receiver_time_override{false};
00288     bool d_active_public_key_id_valid{false};
00289     bool d_new_merkle_tree_loaded{false};
00290     bool d_time_constraint_verified{false};
00291     bool d_kroot_loaded_from_cache{false};
00292
00293     // Provide access to inner functions to Gtest
00294     FRIEND_TEST(OsnmaMsgReceiverTest, TeslaKeyVerification);
00295     FRIEND_TEST(OsnmaMsgReceiverTest, TagVerification);
00296     FRIEND_TEST(OsnmaMsgReceiverTest, TagVerification20Bit);
00297     FRIEND_TEST(OsnmaMsgReceiverTest, TimeConstraintAllowsOnlySlowMac);
00298     FRIEND_TEST(OsnmaMsgReceiverTest, TimeConstraintDecisionIsStoredWithTag);
00299     FRIEND_TEST(OsnmaMsgReceiverTest, ReplayModeSkipsReceiverTimeConstraint);
00300     FRIEND_TEST(OsnmaMsgReceiverTest, TeslaKeyRetainsTimeConstraintDecision);
00301     FRIEND_TEST(OsnmaMsgReceiverTest, TeslaChainResetCanPreserveOrClearDeferredMackQueue);
00302     FRIEND_TEST(OsnmaMsgReceiverTest, PruneOldTeslaKeysDropsExpiredDisclosureKeys);
00303     FRIEND_TEST(OsnmaMsgReceiverTest, TimeConstraintUsesReceiverGuidelineBounds);
00304     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedTransitionHeadersDoNotStartTransitions);
```

```
00305     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedDontUseHeaderDoesNotSkipMackProcessing);
00306     FRIEND_TEST(OsnmaMsgReceiverTest, VerifiedDontUseTagStopsNavigationAuthentication);
00307     FRIEND_TEST(OsnmaMsgReceiverTest, AuthenticatedDontUseClearsDsmAndNavAccumulation);
00308     FRIEND_TEST(OsnmaMsgReceiverTest, StaleKrootStatusSkipsMackProcessing);
00309     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedRevocationHeadersDoNotClearState);
00310     FRIEND_TEST(OsnmaMsgReceiverTest, AuthenticatedCrevPreservesVerifiedFutureKroot);
00311     FRIEND_TEST(OsnmaMsgReceiverTest,
      AuthenticatedRevocationPromotesPreservedFutureKrootAtApplicability);
00312     FRIEND_TEST(OsnmaMsgReceiverTest,
      AuthenticatedPkrevPreservesVerifiedFutureKrootAndCurrentPublicKey);
00313     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedNominalHeaderDoesNotResetTransitionLatches);
00314     FRIEND_TEST(OsnmaMsgReceiverTest, DsmKrootAccumulatorExpiresAfterOneHour);
00315     FRIEND_TEST(OsnmaMsgReceiverTest, DsmPkrAccumulatorExpiresAfterThirteenHours);
00316     FRIEND_TEST(OsnmaMsgReceiverTest, KrootApplicabilitySkipsMackBeforeGst0);
00317     FRIEND_TEST(OsnmaMsgReceiverTest, KrootApplicabilityAcceptsTowhkZero);
00318     FRIEND_TEST(OsnmaMsgReceiverTest, AllZeroPageValidityMaskMeansNoMackPagesAvailable);
00319     FRIEND_TEST(OsnmaMsgReceiverTest, MackBlockWaitsForVerifiedKroot);
00320     FRIEND_TEST(OsnmaMsgReceiverTest, FailedTeslaKeyDoesNotReturnPersistentSuccess);
00321     FRIEND_TEST(OsnmaMsgReceiverTest, OutOfOrderTeslaKeyRejected);
00322     FRIEND_TEST(OsnmaMsgReceiverTest, TeslaKeyLookupUsesWeek);
00323     FRIEND_TEST(OsnmaMsgReceiverTest, FailedDsmPkrKeepsCurrentPublicKey);
00324     FRIEND_TEST(OsnmaMsgReceiverTest, LowerPkidDsmPkrIsRejectedWhilePublicKeyInForce);
00325     FRIEND_TEST(OsnmaMsgReceiverTest, VerifiedDsmPkrSetsActiveKeyIdOnColdStart);
00326     FRIEND_TEST(OsnmaMsgReceiverTest, VerifiedDsmPkrUpdatesPendingKeyIdAcrossRollover);
00327     FRIEND_TEST(OsnmaMsgReceiverTest, MalformedDsmPkrRejectedBeforeLengthCopies);
00328     FRIEND_TEST(OsnmaMsgReceiverTest, KrootWithMismatchedActivePkidIsRejected);
00329     FRIEND_TEST(OsnmaMsgReceiverTest, FailedKrootWithMatchingPkidKeepsActivePublicKey);
00330     FRIEND_TEST(OsnmaMsgReceiverTest, DsmKrootPdkUsesSha256WhenHfSha3);
00331     FRIEND_TEST(OsnmaMsgReceiverTest, VerifiedAlertMessageClearsMerkleMaterial);
00332     FRIEND_TEST(OsnmaMsgReceiverTest, DsmBlockZeroResetsStaleAccumulator);
00333     FRIEND_TEST(OsnmaMsgReceiverTest, DsmBlockZeroKeepsCompatibleUnanchoredBlocks);
00334     FRIEND_TEST(OsnmaMsgReceiverTest, DsmBlockZeroDropsIncompatibleUnanchoredBlocks);
00335     FRIEND_TEST(OsnmaMsgReceiverTest, AnchoredDsmBlockOutsideAnnouncedLengthIsIgnored);
00336     FRIEND_TEST(OsnmaMsgReceiverTest, DsmNonzeroBlockWithDifferentNmaHeaderResetsAnchoredAccumulator);
00337     FRIEND_TEST(OsnmaMsgReceiverTest, RepeatedDsmBlockZeroKeepsPartialAccumulator);
00338     FRIEND_TEST(OsnmaMsgReceiverTest, DsmMessageUsesSavedNmaHeaderForChainRouting);
00339     FRIEND_TEST(OsnmaMsgReceiverTest, FailedKrootDoesNotOverwriteActiveState);
00340     FRIEND_TEST(OsnmaMsgReceiverTest, ChainRenewalDoesNotPromoteUnverifiedKroot);
00341     FRIEND_TEST(OsnmaMsgReceiverTest, ChainRenewalPromotesVerifiedFutureKrootAtApplicability);
00342     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedNominalHeaderDoesNotPromoteVerifiedKroot);
00343     FRIEND_TEST(OsnmaMsgReceiverTest, MalformedKrootRejectedBeforeLengthCopies);
00344     FRIEND_TEST(OsnmaMsgReceiverTest, TeslaKeyVerificationAcrossWeekRollover);
00345     FRIEND_TEST(OsnmaMsgReceiverTest, PendingMackQueueExpiresOldEntries);
00346     FRIEND_TEST(OsnmaMsgReceiverTest, FastTagsWaitForNavDataUntilVerificationWindowExpires);
00347     FRIEND_TEST(OsnmaMsgReceiverTest, GstDeltaSecondsIsSigned);
00348     FRIEND_TEST(OsnmaMsgReceiverTest, UnverifiedNominalHeaderKeepsPublicKeyRenewalPending);
00349     FRIEND_TEST(OsnmaMsgReceiverTest, BuildTagMessageM0);
00350     FRIEND_TEST(OsnmaMsgReceiverTest, NavDataArrivalRetriesPendingTagsImmediately);
00351     FRIEND_TEST(OsnmaMsgReceiverTest, MacseqVerificationUsesMackTime);
00352     FRIEND_TEST(OsnmaMsgReceiverTest, MacseqVerificationMissingMacseqKeepsFixedTagsOnly);
00353     FRIEND_TEST(OsnmaMsgReceiverTest, MacseqVerificationChecksFixedSelfAndCrossSlots);
00354     FRIEND_TEST(OsnmaMsgReceiverTest, MacseqVerificationDiscardsReservedFlexibleTags);
00355     FRIEND_TEST(OsnmaMsgReceiverTest, VerifyPublicKey);
00356     FRIEND_TEST(OsnmaMsgReceiverTest, ComputeBaseLeaf);
00357     FRIEND_TEST(OsnmaMsgReceiverTest, CanonicalPublicKeyFingerprintIncludesPkidNpktAndPoint);
00358     FRIEND_TEST(OsnmaMsgReceiverTest, ComputeMerkleRoot);
00359     FRIEND_TEST(OsnmaMsgReceiverTest, ComputeMerkleRootUsesConfiguredHashFunction);
00360     FRIEND_TEST(OsnmaMsgReceiverTest, DummyCopZeroTagUsesZeroNavigationData);
00361     FRIEND_TEST(OsnmaMsgReceiverTest, NmtDsmPkrWaitsForNewMerkleTree);
00362     FRIEND_TEST(OsnmaMsgReceiverTest, MerkleXmlDoesNotChangeActivePublicKeyType);
00363     FRIEND_TEST(OsnmaMsgReceiverTest, PublicKeyMetadataRestoresPkidForPemHotStart);
00364     FRIEND_TEST(OsnmaMsgReceiverTest, PublicKeyMetadataRejectsMismatchedPemFingerprint);
00365     FRIEND_TEST(OsnmaMsgReceiverTest, NmtDsmPkrUsesCandidateMerkleTree);
00366     FRIEND_TEST(OsnmaMsgReceiverTest, UnsupportedKrootParametersAreRejected);
00367     FRIEND_TEST(OsnmaMsgReceiverTest, UnsupportedKrootParametersSkipMackProcessing);
00368     FRIEND_TEST(OsnmaMsgReceiverTest, MackWithMissingTag0StillParsesMackBody);
00369     FRIEND_TEST(OsnmaMsgReceiverTest, PartialMackKeepsAvailableFixedTags);
00370     FRIEND_TEST(OsnmaMsgReceiverTest, PartialTeslaKeyReconstructedAcrossSatellites);
00371     FRIEND_TEST(OsnmaMsgReceiverTest, CachedKrootWithoutMetadataIsNotPromotedAtStartup);
00372     FRIEND_TEST(OsnmaMsgReceiverTest, CachedKrootMetadataStoresFreshnessAndApplicability);
00373     FRIEND_TEST(OsnmaMsgReceiverTest, StaleCachedKrootMetadataIsRejected);
00374     friend class OsnmaTestVectors;
00375     FRIEND_TEST(OsnmaTestVectors, NominalTestConf1);
00376     FRIEND_TEST(OsnmaTestVectors, NominalTestConf2);
00377     FRIEND_TEST(OsnmaTestVectors, PublicKeyRenewal);
00378     FRIEND_TEST(OsnmaTestVectors, PublicKeyRevocation);
00379     FRIEND_TEST(OsnmaTestVectors, ChainRenewal);
00380     FRIEND_TEST(OsnmaTestVectors, ChainRevocation);
00381     FRIEND_TEST(OsnmaTestVectors, AlertMessage);
00382 };
00383
00384
00385 /** \} */
00386 /** \} */
00387 #endif  // GNSS_SDR_OSNMA_MSG_RECEIVER_H
```

13.598 osnma_nav_data_manager.h File Reference

Class for Galileo OSNMA navigation data management.

```
#include "osnma_data.h"
#include <cstdint>
#include <map>
#include <string>
#include <vector>
```

Classes

- class OSNMA_NavDataManager

  Class for managing OSNMA navigation data.

13.598.1 Detailed Description

Class for Galileo OSNMA navigation data management.

Author

Cesare Ghionoiu-Martinez, 2020-2023 cesare.martinez(at)proton.me

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file osnma_nav_data_manager.h.

13.599 osnma_nav_data_manager.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_nav_data_manager.h
00003  * \brief Class for Galileo OSNMA navigation data management
00004  * \author Cesare Ghionoiu-Martinez, 2020-2023 cesare.martinez(at)proton.me
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_OSNMA_NAV_DATA_MANAGER_H
00018 #define GNSS_SDR_OSNMA_NAV_DATA_MANAGER_H
00019
00020 #include "osnma_data.h"  // for OSNMA_NavData, Tag
00021 #include <cstdint>       // for uint32_t
00022 #include <map>
00023 #include <string>
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup Core_Receiver_Library
00029  * \{ */
00030
00031 /**
00032  * @class OSNMA_NavDataManager
00033  * @brief Class for managing OSNMA navigation data
00034  */
00035 class OSNMA_NavDataManager
00036 {
00037 public:
00038     OSNMA_NavDataManager() = default;
00039
00040     void log_status() const;
00041     bool have_nav_data(const Tag& t) const;
00042     std::string get_navigation_data(const Tag& t) const;
00043
00044     void add_navigation_data(const std::string& nav_bits, uint32_t PRNd, uint32_t WN, uint32_t TOW);
```

```
00045     void update_nav_data(const std::multimap<uint32_t, Tag>& tags_verified, uint8_t tag_size);
00046     void reset_tag_accumulations();
00047     void prune_old_navigation_data(uint32_t WN, uint32_t TOW);
00048     bool have_nav_data(const std::string& nav_bits, uint32_t PRNd, uint32_t WN, uint32_t TOW);
00049     std::vector<OSNMA_NavData> get_verified_data();
00050
00051 private:
00052     std::map<uint32_t, std::map<uint64_t, OSNMA_NavData» d_satellite_nav_data{};  // NavData sorted by
    [PRNd][GST_start_seconds]
00053     const uint32_t L_t_min{40};
00054     const uint16_t EPH_SIZE{549};
00055     const uint16_t UTC_SIZE{141};
00056 };
00057
00058 /** \} */
00059 /** \} */
00060 #endif  // GNSS_SDR_OSNMA_NAV_DATA_MANAGER_H
```

13.600 protobuf_cleanup_manager.h File Reference

Singleton class that implements the shared resource cleaner pattern for the Protocol Buffers library.

`#include <google/protobuf/message_lite.h>`

Classes

- class Protobuf_Cleanup_Manager

  This class implements the Protocol Buffers library shared memory cleanup manager.

13.600.1 Detailed Description

Singleton class that implements the shared resource cleaner pattern for the Protocol Buffers library.

Author

Vladislav P, 2026. vladisslav2011(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file protobuf_cleanup_manager.h.

13.601 protobuf_cleanup_manager.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file protobuf_cleanup_manager.h
00003  * \brief Singleton class that implements the shared resource cleaner pattern
00004  * for the Protocol Buffers library
00005  * \author Vladislav P, 2026. vladisslav2011(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_PROTOBUF_CLEANUP_MANAGER_H
00019 #define GNSS_SDR_PROTOBUF_CLEANUP_MANAGER_H
00020
00021 #include <google/protobuf/message_lite.h>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup Core_Receiver_Library
00026  * \{ */
00027
00028 /*!
00029  * \brief This class implements the Protocol Buffers library shared memory cleanup manager.
```

```
00030  *
00031  * google::protobuf::ShutdownProtobufLibrary() must be called only once, at the end
00032  * of the program execution: it is not safe to use any other part of the library
00033  * after that call. Calling it earlier (e.g., from the destructor of a class that
00034  * uses protobuf) results in use-after-free memory corruption and random crashes.
00035  * Call Protobuf_Cleanup_Manager::get() in the constructor of every class that
00036  * uses the protobuf library, so that google::protobuf::ShutdownProtobufLibrary()
00037  * is called at program shutdown, when the singleton is destroyed.
00038  */
00039 class Protobuf_Cleanup_Manager
00040 {
00041 public:
00042     static Protobuf_Cleanup_Manager& get()
00043     {
00044         static Protobuf_Cleanup_Manager instance{};
00045         return instance;
00046     }
00047
00048     Protobuf_Cleanup_Manager(const Protobuf_Cleanup_Manager&) = delete;
00049     Protobuf_Cleanup_Manager& operator=(const Protobuf_Cleanup_Manager&) = delete;
00050     Protobuf_Cleanup_Manager(Protobuf_Cleanup_Manager&&) = delete;
00051     Protobuf_Cleanup_Manager& operator=(Protobuf_Cleanup_Manager&&) = delete;
00052
00053 private:
00054     Protobuf_Cleanup_Manager() = default;
00055     ~Protobuf_Cleanup_Manager()
00056     {
00057         google::protobuf::ShutdownProtobufLibrary();
00058     }
00059 };
00060
00061 /** \} */
00062 /** \} */
00063 #endif  // GNSS_SDR_PROTOBUF_CLEANUP_MANAGER_H
```

13.602 serdes_nav_message.h File Reference

Serialization / Deserialization of Nav_Message_Packet objects using Protocol Buffers.

```
#include "nav_message.pb.h"
#include "nav_message_packet.h"
#include "protobuf_cleanup_manager.h"
#include <cstdint>
#include <memory>
#include <string>
#include <utility>
```

Classes

- class Serdes_Nav_Message

  This class implements serialization and deserialization of Nav_Message_Packet objects using Protocol Buffers.

13.602.1 Detailed Description

Serialization / Deserialization of Nav_Message_Packet objects using Protocol Buffers.

Author

Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file serdes_nav_message.h.

13.603 serdes_nav_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file serdes_nav_message.h
00003  * \brief Serialization / Deserialization of Nav_Message_Packet objects using
```

```
00004  * Protocol Buffers
00005  * \author Carles Fernandez-Prades, 2021. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SERDES_NAV_MESSAGE_H
00019 #define GNSS_SDR_SERDES_NAV_MESSAGE_H
00020
00021 #include "nav_message.pb.h"  // file created by Protocol Buffers at compile time
00022 #include "nav_message_packet.h"
00023 #include "protobuf_cleanup_manager.h"
00024 #include <cstdint>
00025 #include <memory>
00026 #include <string>
00027 #include <utility>
00028
00029 /** \addtogroup Telemetry_Decoder
00030  * \{ */
00031 /** \addtogroup Telemetry_Decoder_libs
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief This class implements serialization and deserialization of
00037  * Nav_Message_Packet objects using Protocol Buffers.
00038  */
00039 class Serdes_Nav_Message
00040 {
00041 public:
00042     Serdes_Nav_Message()
00043     {
00044         // Verify that the version of the library that we linked against is
00045         // compatible with the version of the headers we compiled against.
00046         GOOGLE_PROTOBUF_VERIFY_VERSION;
00047
00048         // Make sure google::protobuf::ShutdownProtobufLibrary() is called only once,
00049         // at the end of the program execution (see Protobuf_Cleanup_Manager)
00050         Protobuf_Cleanup_Manager::get();
00051     }
00052
00053     inline Serdes_Nav_Message(const Serdes_Nav_Message& other) noexcept : navmsg_(other.navmsg_)  //!<
    Copy constructor
00054     {
00055     }
00056
00057     inline Serdes_Nav_Message& operator=(const Serdes_Nav_Message& rhs) noexcept  //!< Copy assignment
    operator
00058     {
00059         if (this != &rhs)
00060             {
00061                 this->navmsg_.CopyFrom(rhs.navmsg_);
00062             }
00063         return *this;
00064     }
00065
00066     inline Serdes_Nav_Message(Serdes_Nav_Message&& other) noexcept : navmsg_(std::move(other.navmsg_))
    //!< Move constructor
00067     {
00068         // Set the other object's navmsg_ to a default-constructed state
00069         other.navmsg_ = gnss_sdr::navMsg{};
00070     }
00071
00072     inline Serdes_Nav_Message& operator=(Serdes_Nav_Message&& other) noexcept  //!< Move assignment
    operator
00073     {
00074         if (this != &other)
00075             {
00076                 navmsg_ = std::move(other.navmsg_);
00077                 other.navmsg_ = gnss_sdr::navMsg{};
00078             }
00079         return *this;
00080     }
00081
00082     inline std::string createProtobuffer(const std::shared_ptr<Nav_Message_Packet> nav_msg_packet)
    //!< Serialization into a string
00083     {
00084         navmsg_.Clear();
00085         std::string data;
```

```
00086
00087         navmsg_.set_system(nav_msg_packet->system);
00088         navmsg_.set_signal(nav_msg_packet->signal);
00089         navmsg_.set_prn(nav_msg_packet->prn);
00090         navmsg_.set_tow_at_current_symbol_ms(nav_msg_packet->tow_at_current_symbol_ms);
00091         navmsg_.set_nav_message(nav_msg_packet->nav_message);
00092
00093         if (!navmsg_.SerializeToString(&data))
00094             {
00095                 return {};
00096             }
00097
00098         return data;
00099     }
00100
00101     inline Nav_Message_Packet readProtobuffer(const gnss_sdr::navMsg& msg) const  //!< Deserialization
00102     {
00103         Nav_Message_Packet navmsg;
00104
00105         navmsg.system = msg.system();
00106         navmsg.signal = msg.signal();
00107         navmsg.prn = msg.prn();
00108         navmsg.tow_at_current_symbol_ms = msg.tow_at_current_symbol_ms();
00109         navmsg.nav_message = msg.nav_message();
00110
00111         return navmsg;
00112     }
00113
00114 private:
00115     gnss_sdr::navMsg navmsg_{};
00116 };
00117
00118
00119 /** \} */
00120 /** \} */
00121 #endif  // GNSS_SDR_SERDES_NAV_MESSAGE_H
```

13.604 string_converter.h File Reference

Interface of a class that interprets the contents of a string and converts it into different types.

```
#include <cstdint>
#include <string>
```

Classes

- class StringConverter

  Class that interprets the contents of a string and converts it into different types.

13.604.1 Detailed Description

Interface of a class that interprets the contents of a string and converts it into different types.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file string_converter.h.

13.605 string_converter.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file string_converter.h
00003  * \brief Interface of a class that interprets the contents of a string
00004  * and converts it into different types.
00005  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_STRING_CONVERTER_H
00020 #define GNSS_SDR_STRING_CONVERTER_H
00021
00022 #include <cstdint>
00023 #include <string>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup Core_Receiver_Library
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief Class that interprets the contents of a string
00033  * and converts it into different types.
00034  */
00035 class StringConverter
00036 {
00037 public:
00038     StringConverter() = default;
00039     ~StringConverter() = default;
00040
00041     bool convert(const std::string& value, bool default_value);
00042     int64_t convert(const std::string& value, int64_t default_value);
00043     uint64_t convert(const std::string& value, uint64_t default_value);
00044     int32_t convert(const std::string& value, int32_t default_value);
00045     uint32_t convert(const std::string& value, uint32_t default_value);
00046     int16_t convert(const std::string& value, int16_t default_value);
00047     uint16_t convert(const std::string& value, uint16_t default_value);
00048     float convert(const std::string& value, float default_value);
00049     double convert(const std::string& value, double default_value);
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_STRING_CONVERTER_H
```

13.606 uio_fpga.h File Reference

This library contains functions to determine the uio device driver file that corresponds to a hardware accelerator device name in the FPGA.

```
#include <cstdint>
#include <string>
```

Functions

- const std::string **uio_dir** ("/sys/class/uio/")
- const std::string **uio_filename** ("uio")
- const std::string **uio_subdir_name** ("/name")
- int32_t find_uio_dev_file_name (std::string &device_file_name, const std::string &device_name, uint32_t device_num)

  This function finds the uio device driver device file name out of the device name and the device number.
- uint32_t get_num_devices (const std::string &device_name)

  Returns the number of devices (tracking multicorrelators) of type device_name available in the FPGA.

13.606.1 Detailed Description

This library contains functions to determine the uio device driver file that corresponds to a hardware accelerator device name in the FPGA.

Author

Marc Majoral, 2020. mmajoral(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file uio_fpga.h.

13.607 uio_fpga.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file uio_fpga.h
00003  * \brief This library contains functions to determine the uio device driver
00004  * file that corresponds to a hardware accelerator device name in the FPGA.
00005  * \author Marc Majoral, 2020. mmajoral(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_UIO_FPGA_H
00019 #define GNSS_SDR_UIO_FPGA_H
00020
00021 #include <cstdint>
00022 #include <string>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup Core_Receiver_Library
00027  * \{ */
00028
00029 const std::string uio_dir("/sys/class/uio/");
00030 const std::string uio_filename("uio");
00031 const std::string uio_subdir_name("/name");
00032
00033 /*!
00034  * \brief This function finds the uio device driver device file name out of the
00035  * device name and the device number.
00036  */
00037 int32_t find_uio_dev_file_name(std::string &device_file_name,
00038     const std::string &device_name,
00039     uint32_t device_num);
00040
00041 /*!
00042  * \brief Returns the number of devices (tracking multicorrelators) of type device_name available in
      the FPGA
00043  */
00044 uint32_t get_num_devices(const std::string &device_name);
00045
00046 /** \} */
00047 /** \} */
00048 #endif  // GNSS_SDR_UIO_FPGA_H
```

13.608 gnss_synchro_monitor.h File Reference

Interface of a receiver monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.

```
#include "gnss_block_interface.h"
#include "gnss_synchro_udp_sink.h"
#include <gnuradio/block.h>
#include <gnuradio/runtime_types.h>
#include <memory>
#include <string>
#include <vector>
```

Classes

- class gnss_synchro_monitor

  This class implements a monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.

Typedefs

- using gnss_synchro_monitor_sptr = gnss_shared_ptr<gnss_synchro_monitor>

Functions

- gnss_synchro_monitor_sptr **gnss_synchro_make_monitor** (int n_channels, int decimation_factor, const std::vector< std::string > &udp_ports, const std::vector< std::string > &udp_addresses, bool enable_↩ protobuf)

13.608.1 Detailed Description

Interface of a receiver monitoring block which allows sending a data stream with the receiver internal parameters (Gnss_Synchro objects) to local or remote clients over UDP.

Author

Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_synchro_monitor.h.

13.609 gnss_synchro_monitor.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_synchro_monitor.h
00003  * \brief Interface of a receiver monitoring block which allows sending
00004  * a data stream with the receiver internal parameters (Gnss_Synchro objects)
00005  * to local or remote clients over UDP.
00006  *
00007  * \author Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_GNSS_SYNCHRO_MONITOR_H
00021 #define GNSS_SDR_GNSS_SYNCHRO_MONITOR_H
00022
00023 #include "gnss_block_interface.h"
00024 #include "gnss_synchro_udp_sink.h"
00025 #include <gnuradio/block.h>
00026 #include <gnuradio/runtime_types.h>  // for gr_vector_void_star
00027 #include <memory>
00028 #include <string>
00029 #include <vector>
00030
00031 /** \addtogroup Core
00032  * \{ */
00033 /** \addtogroup Gnss_Synchro_Monitor core_monitor
00034  * Classes for the Gnss_Synchro monitor.
00035  * \{ */
00036
00037
00038 class gnss_synchro_monitor;
00039
00040 using gnss_synchro_monitor_sptr = gnss_shared_ptr<gnss_synchro_monitor>;
00041
```

```
00042 gnss_synchro_monitor_sptr gnss_synchro_make_monitor(int n_channels,
00043     int decimation_factor,
00044     const std::vector<std::string>& udp_ports,
00045     const std::vector<std::string>& udp_addresses,
00046     bool enable_protobuf);
00047
00048 /*!
00049  * \brief This class implements a monitoring block which allows sending
00050  * a data stream with the receiver internal parameters (Gnss_Synchro objects)
00051  * to local or remote clients over UDP.
00052  */
00053 class gnss_synchro_monitor : public gr::block
00054 {
00055 public:
00056     ~gnss_synchro_monitor() = default;  //!< Default destructor
00057     void forecast(int noutput_items, gr_vector_int& ninput_items_required);
00058     int general_work(int noutput_items, gr_vector_int& ninput_items,
00059         gr_vector_const_void_star& input_items, gr_vector_void_star& output_items);
00060
00061 private:
00062     friend gnss_synchro_monitor_sptr gnss_synchro_make_monitor(int n_channels,
00063         int decimation_factor,
00064         const std::vector<std::string>& udp_ports,
00065         const std::vector<std::string>& udp_addresses,
00066         bool enable_protobuf);
00067
00068     gnss_synchro_monitor(int n_channels,
00069         int decimation_factor,
00070         const std::vector<std::string>& udp_ports,
00071         const std::vector<std::string>& udp_addresses,
00072         bool enable_protobuf);
00073
00074     std::unique_ptr<Gnss_Synchro_Udp_Sink> udp_sink_ptr;
00075     int count;
00076     int d_nchannels;
00077     int d_decimation_factor;
00078 };
00079
00080
00081 /** \} */
00082 /** \} */
00083 #endif  // GNSS_SDR_GNSS_SYNCHRO_MONITOR_H
```

13.610 gnss_synchro_udp_sink.h File Reference

Interface of a class that sends serialized Gnss_Synchro objects over udp to one or multiple endponits.

```
#include "gnss_synchro.h"
#include "serdes_gnss_synchro.h"
#include <boost/asio.hpp>
#include <boost/system/error_code.hpp>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class Gnss_Synchro_Udp_Sink

  This class sends serialized Gnss_Synchro objects over UDP to one or multiple endpoints.

Typedefs

- using b_io_context = boost::asio::io_service

13.610.1 Detailed Description

Interface of a class that sends serialized Gnss_Synchro objects over udp to one or multiple endponits.

Author

Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_synchro_udp_sink.h.

13.611 gnss_synchro_udp_sink.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_synchro_udp_sink.h
00003  * \brief Interface of a class that sends serialized Gnss_Synchro objects
00004  * over udp to one or multiple endponits
00005  * \author Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GNSS_SYNCHRO_UDP_SINK_H
00019 #define GNSS_SDR_GNSS_SYNCHRO_UDP_SINK_H
00020
00021 #include "gnss_synchro.h"
00022 #include "serdes_gnss_synchro.h"
00023 #include <boost/asio.hpp>
00024 #include <boost/system/error_code.hpp>
00025 #include <cstdint>
00026 #include <string>
00027 #include <vector>
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup Gnss_Synchro_Monitor
00032  * \{ */
00033
00034
00035 #if USE_BOOST_ASIO_IO_CONTEXT
00036 using b_io_context = boost::asio::io_context;
00037 #else
00038 using b_io_context = boost::asio::io_service;
00039 #endif
00040
00041 /*!
00042  * \brief This class sends serialized Gnss_Synchro objects
00043  * over UDP to one or multiple endpoints.
00044  */
00045 class Gnss_Synchro_Udp_Sink
00046 {
00047 public:
00048     Gnss_Synchro_Udp_Sink(const std::vector<std::string>& addresses, const std::vector<std::string>&
      ports, bool enable_protobuf);
00049     bool write_gnss_synchro(const std::vector<Gnss_Synchro>& stocks);
00050
00051 private:
00052     b_io_context io_context;
00053     boost::asio::ip::udp::socket socket;
00054     boost::system::error_code error;
00055     std::vector<boost::asio::ip::udp::endpoint> endpoints;
00056     Serdes_Gnss_Synchro serdes;
00057     bool use_protobuf;
00058 };
00059
00060
00061 /** \} */
00062 /** \} */
00063 #endif  // GNSS_SDR_GNSS_SYNCHRO_UDP_SINK_H
```

13.612 serdes_gnss_synchro.h File Reference

Serialization / Deserialization of Gnss_Synchro objects using Protocol Buffers.

```
#include "gnss_synchro.h"
#include "gnss_synchro.pb.h"
#include "protobuf_cleanup_manager.h"
#include <array>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Serdes_Gnss_Synchro

  This class implements serialization and deserialization of Gnss_Synchro objects using Protocol Buffers.

13.612.1 Detailed Description

Serialization / Deserialization of Gnss_Synchro objects using Protocol Buffers.

Author

Carles Fernandez-Prades, 2019. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file serdes_gnss_synchro.h.

13.613 serdes_gnss_synchro.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file serdes_gnss_synchro.h
00003  * \brief Serialization / Deserialization of Gnss_Synchro objects using
00004  * Protocol Buffers
00005  * \author Carles Fernandez-Prades, 2019. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_SERDES_GNSS_SYNCHRO_H
00019 #define GNSS_SDR_SERDES_GNSS_SYNCHRO_H
00020
00021 #include "gnss_synchro.h"
00022 #include "gnss_synchro.pb.h"  // file created by Protocol Buffers at compile time
00023 #include "protobuf_cleanup_manager.h"
00024 #include <array>
00025 #include <string>
00026 #include <utility>
00027 #include <vector>
00028
00029 /*!
00030  * \brief This class implements serialization and deserialization of
00031  * Gnss_Synchro objects using Protocol Buffers.
00032  */
00033 class Serdes_Gnss_Synchro
00034 {
00035 public:
00036     Serdes_Gnss_Synchro()
00037     {
00038         // Verify that the version of the library that we linked against is
00039         // compatible with the version of the headers we compiled against.
00040         GOOGLE_PROTOBUF_VERIFY_VERSION;
00041
00042         // Make sure google::protobuf::ShutdownProtobufLibrary() is called only once,
00043         // at the end of the program execution (see Protobuf_Cleanup_Manager)
00044         Protobuf_Cleanup_Manager::get();
00045     }
```

```
00046
00047     inline Serdes_Gnss_Synchro(const Serdes_Gnss_Synchro& other) noexcept  //!< Copy constructor
00048     {
00049         this->observables = other.observables;
00050     }
00051
00052     inline Serdes_Gnss_Synchro& operator=(const Serdes_Gnss_Synchro& rhs) noexcept  //!< Copy
      assignment operator
00053     {
00054         this->observables = rhs.observables;
00055         return *this;
00056     }
00057
00058     inline Serdes_Gnss_Synchro(Serdes_Gnss_Synchro&& other) noexcept  //!< Move constructor
00059     {
00060         this->observables = std::move(other.observables);
00061     }
00062
00063     inline Serdes_Gnss_Synchro& operator=(Serdes_Gnss_Synchro&& other) noexcept  //!< Move assignment
      operator
00064     {
00065         if (this != &other)
00066             {
00067                 this->observables = std::move(other.observables);
00068             }
00069         return *this;
00070     }
00071
00072     inline std::string createProtobuffer(const std::vector<Gnss_Synchro>& vgs)  //!< Serialization
      into a string
00073     {
00074         observables.Clear();
00075         std::string data;
00076         for (const auto& gs : vgs)
00077             {
00078                 gnss_sdr::GnssSynchro* obs = observables.add_observable();
00079                 char c = gs.System;
00080                 const std::string sys(1, c);
00081
00082                 std::array<char, 2> cc;
00083                 cc[0] = gs.Signal[0];
00084                 cc[1] = gs.Signal[1];
00085                 const std::string sig(cc.cbegin(), cc.cend());
00086
00087                 obs->set_system(sys);
00088                 obs->set_signal(sig);
00089                 obs->set_prn(gs.PRN);
00090                 obs->set_channel_id(gs.Channel_ID);
00091
00092                 obs->set_acq_delay_samples(gs.Acq_delay_samples);
00093                 obs->set_acq_doppler_hz(gs.Acq_doppler_hz);
00094                 obs->set_acq_samplestamp_samples(gs.Acq_samplestamp_samples);
00095                 obs->set_acq_doppler_step(gs.Acq_doppler_step);
00096                 obs->set_flag_valid_acquisition(gs.Flag_valid_acquisition);
00097
00098                 obs->set_fs(gs.fs);
00099                 obs->set_prompt_i(gs.Prompt_I);
00100                 obs->set_prompt_q(gs.Prompt_Q);
00101                 obs->set_cn0_db_hz(gs.CN0_dB_hz);
00102                 obs->set_carrier_doppler_hz(gs.Carrier_Doppler_hz);
00103                 obs->set_carrier_phase_rads(gs.Carrier_phase_rads);
00104                 obs->set_code_phase_samples(gs.Code_phase_samples);
00105                 obs->set_tracking_sample_counter(gs.Tracking_sample_counter);
00106                 obs->set_flag_valid_symbol_output(gs.Flag_valid_symbol_output);
00107                 obs->set_correlation_length_ms(gs.correlation_length_ms);
00108
00109                 obs->set_flag_valid_word(gs.Flag_valid_word);
00110                 obs->set_tow_at_current_symbol_ms(gs.TOW_at_current_symbol_ms);
00111
00112                 obs->set_pseudorange_m(gs.Pseudorange_m);
00113                 obs->set_rx_time(gs.RX_time);
00114                 obs->set_flag_valid_pseudorange(gs.Flag_valid_pseudorange);
00115                 obs->set_flag_pll_180_deg_phase_locked(gs.Flag_PLL_180_deg_phase_locked);
00116                 obs->set_flag_cycle_slip(gs.Flag_cycle_slip);
00117                 obs->set_flag_carrier_phase_discontinuity(!gs.Flag_carrier_phase_continuous);
00118                 obs->set_flag_half_cycle_slip(gs.Flag_half_cycle_slip);
00119                 obs->set_interp_tow_ms(gs.interp_TOW_ms);
00120             }
00121         if (!observables.SerializeToString(&data))
00122             {
00123                 return {};
00124             }
00125         return data;
00126     }
00127
00128     inline std::vector<Gnss_Synchro> readProtobuffer(const gnss_sdr::Observables& obs) const  //!<
      Deserialization
```

```
00129     {
00130         std::vector<Gnss_Synchro> vgs;
00131         vgs.reserve(obs.observable_size());
00132         for (int i = 0; i < obs.observable_size(); ++i)
00133             {
00134                 const gnss_sdr::GnssSynchro& gs_read = obs.observable(i);
00135                 Gnss_Synchro gs = Gnss_Synchro();
00136                 gs.System = gs_read.system()[0];
00137                 gs.Signal[0] = gs_read.signal()[0];
00138                 gs.Signal[1] = gs_read.signal()[1];
00139                 gs.Signal[2] = '\0';
00140                 gs.PRN = gs_read.prn();
00141                 gs.Channel_ID = gs_read.channel_id();
00142
00143                 gs.Acq_delay_samples = gs_read.acq_delay_samples();
00144                 gs.Acq_doppler_hz = gs_read.acq_doppler_hz();
00145                 gs.Acq_samplestamp_samples = gs_read.acq_samplestamp_samples();
00146                 gs.Acq_doppler_step = gs_read.acq_doppler_step();
00147                 gs.Flag_valid_acquisition = gs_read.flag_valid_acquisition();
00148
00149                 gs.fs = gs_read.fs();
00150                 gs.Prompt_I = gs_read.prompt_i();
00151                 gs.Prompt_Q = gs_read.prompt_q();
00152                 gs.CN0_dB_hz = gs_read.cn0_db_hz();
00153                 gs.Carrier_Doppler_hz = gs_read.carrier_doppler_hz();
00154                 gs.Carrier_phase_rads = gs_read.carrier_phase_rads();
00155                 gs.Code_phase_samples = gs_read.code_phase_samples();
00156                 gs.Tracking_sample_counter = gs_read.tracking_sample_counter();
00157                 gs.Flag_valid_symbol_output = gs_read.flag_valid_symbol_output();
00158                 gs.correlation_length_ms = gs_read.correlation_length_ms();
00159
00160                 gs.Flag_valid_word = gs_read.flag_valid_word();
00161                 gs.TOW_at_current_symbol_ms = gs_read.tow_at_current_symbol_ms();
00162
00163                 gs.Pseudorange_m = gs_read.pseudorange_m();
00164                 gs.RX_time = gs_read.rx_time();
00165                 gs.Flag_valid_pseudorange = gs_read.flag_valid_pseudorange();
00166                 gs.Flag_PLL_180_deg_phase_locked = gs_read.flag_pll_180_deg_phase_locked();
00167                 gs.Flag_cycle_slip = gs_read.flag_cycle_slip();
00168                 gs.Flag_carrier_phase_continuous = !gs_read.flag_carrier_phase_discontinuity();
00169                 gs.Flag_half_cycle_slip = gs_read.flag_half_cycle_slip();
00170                 gs.interp_TOW_ms = gs_read.interp_tow_ms();
00171
00172                 vgs.push_back(gs);
00173             }
00174         return vgs;
00175     }
00176
00177 private:
00178     gnss_sdr::Observables observables{};
00179 };
00180
00181 #endif  // GNSS_SDR_SERDES_GNSS_SYNCHRO_H
```

13.614 concurrent_map.h File Reference

Interface of a thread-safe std::map.

```
#include <map>
#include <mutex>
#include <utility>
```

Classes

- class Concurrent_Map< Data >

  This class implements a thread-safe std::map.

13.614.1 Detailed Description

Interface of a thread-safe std::map.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file concurrent_map.h.

13.615 concurrent_map.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file concurrent_map.h
00003  * \brief Interface of a thread-safe std::map
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_CONCURRENT_MAP_H
00018 #define GNSS_SDR_CONCURRENT_MAP_H
00019
00020 #include <map>
00021 #include <mutex>
00022 #include <utility>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup Core_Receiver core_receiver
00027  * \{ */
00028
00029
00030 template <typename Data>
00031
00032
00033 /*!
00034  * \brief This class implements a thread-safe std::map
00035  *
00036  */
00037 class Concurrent_Map
00038 {
00039 public:
00040     void write(int key, Data const& data)
00041     {
00042         std::lock_guard<std::mutex> lock(the_mutex);
00043         auto data_iter = the_map.find(key);
00044         if (data_iter != the_map.end())
00045             {
00046                 data_iter->second = data;  // update
00047             }
00048         else
00049             {
00050                 the_map.insert(std::pair<int, Data>(key, data));  // insert does not overwrite if the
      item already exists in the map!
00051             }
00052     }
00053
00054     std::map<int, Data> get_map_copy() const&
00055     {
00056         std::lock_guard<std::mutex> lock(the_mutex);
00057         return the_map;  // This implicitly creates a copy
00058     }
00059
00060     std::map<int, Data> get_map_copy() &&
00061     {
00062         std::lock_guard<std::mutex> lock(the_mutex);
00063         return std::move(the_map);
00064     }
00065
00066     size_t size() const
00067     {
00068         std::lock_guard<std::mutex> lock(the_mutex);
00069         return the_map.size();
00070     }
00071
```

```
00072     bool read(int key, Data& p_data) const
00073     {
00074         std::lock_guard<std::mutex> lock(the_mutex);
00075         auto data_iter = the_map.find(key);
00076         if (data_iter != the_map.end())
00077             {
00078                 p_data = data_iter->second;
00079                 return true;
00080             }
00081         return false;
00082     }
00083
00084 private:
00085     std::map<int, Data> the_map;
00086     mutable std::mutex the_mutex;
00087 };
00088
00089
00090 /** \} */
00091 /** \} */
00092 #endif  // GNSS_SDR_CONCURRENT_MAP_H
```

13.616 concurrent_queue.h File Reference

Interface of a thread-safe std::queue.

```
#include <chrono>
#include <condition_variable>
#include <cstddef>
#include <mutex>
#include <queue>
#include <utility>
```

Classes

- class Concurrent_Queue< Data >

  This class implements a thread-safe std::queue.

13.616.1 Detailed Description

Interface of a thread-safe std::queue.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file concurrent_queue.h.

13.617 concurrent_queue.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file concurrent_queue.h
00003  * \brief Interface of a thread-safe std::queue
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_CONCURRENT_QUEUE_H
00018 #define GNSS_SDR_CONCURRENT_QUEUE_H
```

```
00019
00020 #include <chrono>
00021 #include <condition_variable>
00022 #include <cstddef>
00023 #include <mutex>
00024 #include <queue>
00025 #include <utility>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup Core_Receiver
00030  * \{ */
00031
00032
00033 template <typename Data>
00034
00035 /*!
00036  * \brief This class implements a thread-safe std::queue
00037  */
00038 class Concurrent_Queue
00039 {
00040 public:
00041     void push(const Data& data)
00042     {
00043         {
00044             std::lock_guard<std::mutex> lock(the_mutex);
00045             the_queue.push(data);
00046         }
00047         the_condition_variable.notify_one();
00048     }
00049
00050     void push(Data&& data)
00051     {
00052         {
00053             std::lock_guard<std::mutex> lock(the_mutex);
00054             the_queue.push(std::move(data));
00055         }
00056         the_condition_variable.notify_one();
00057     }
00058
00059     bool empty() const noexcept
00060     {
00061         return size() == 0;
00062     }
00063
00064     size_t size() const noexcept
00065     {
00066         std::lock_guard<std::mutex> lock(the_mutex);
00067         return the_queue.size();
00068     }
00069
00070     void clear()
00071     {
00072         std::lock_guard<std::mutex> lock(the_mutex);
00073         std::queue<Data>().swap(the_queue);
00074     }
00075
00076     bool try_pop(Data& popped_value)
00077     {
00078         std::lock_guard<std::mutex> lock(the_mutex);
00079         if (the_queue.empty())
00080             {
00081                 return false;
00082             }
00083         popped_value = std::move(the_queue.front());
00084         the_queue.pop();
00085         return true;
00086     }
00087
00088     void wait_and_pop(Data& popped_value)
00089     {
00090         std::unique_lock<std::mutex> lock(the_mutex);
00091         the_condition_variable.wait(lock, [this] { return !the_queue.empty(); });
00092         popped_value = std::move(the_queue.front());
00093         the_queue.pop();
00094     }
00095
00096     bool timed_wait_and_pop(Data& popped_value, int wait_ms)
00097     {
00098         std::unique_lock<std::mutex> lock(the_mutex);
00099         if (!the_condition_variable.wait_for(lock,
00100                 std::chrono::milliseconds(wait_ms),
00101                 [this] { return !the_queue.empty(); }))
00102             {
00103                 return false;
00104             }
00105         popped_value = std::move(the_queue.front());
```

```
00106         the_queue.pop();
00107         return true;
00108     }
00109
00110 private:
00111     std::queue<Data> the_queue;
00112     mutable std::mutex the_mutex;
00113     std::condition_variable the_condition_variable;
00114 };
00115
00116
00117 /** \} */
00118 /** \} */
00119 #endif  // GNSS_SDR_CONCURRENT_QUEUE_H
```

13.618 control_thread.h File Reference

Interface of the receiver control plane.

```
#include "agnss_ref_location.h"
#include "agnss_ref_time.h"
#include "channel_event.h"
#include "command_event.h"
#include "concurrent_queue.h"
#include "gnss_sdr_supl_client.h"
#include "tcp_cmd_interface.h"
#include <pmt/pmt.h>
#include <array>
#include <cstddef>
#include <cstdint>
#include <memory>
#include <string>
#include <thread>
#include <typeinfo>
#include <utility>
#include <vector>
```

Classes

- class ControlThread

  This class represents the main thread of the application, so the name is ControlThread. This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions.

13.618.1 Detailed Description

Interface of the receiver control plane.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

GNSS Receiver Control Plane: connects the flowgraph, starts running it, and while it does not stop, reads the control messages generated by the blocks, processes them, and applies the corresponding actions.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file control_thread.h.

13.619 control_thread.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file control_thread.h
```

```
00003  * \brief Interface of the receiver control plane
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * GNSS Receiver Control Plane: connects the flowgraph, starts running it,
00007  * and while it does not stop, reads the control messages generated by the blocks,
00008  * processes them, and applies the corresponding actions.
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021 #ifndef GNSS_SDR_CONTROL_THREAD_H
00022 #define GNSS_SDR_CONTROL_THREAD_H
00023
00024 #include "agnss_ref_location.h"    // for Agnss_Ref_Location
00025 #include "agnss_ref_time.h"        // for Agnss_Ref_Time
00026 #include "channel_event.h"         // for channel_event_sptr
00027 #include "command_event.h"         // for command_event_sptr
00028 #include "concurrent_queue.h"      // for Concurrent_Queue
00029 #include "gnss_sdr_supl_client.h"  // for Gnss_Sdr_Supl_Client
00030 #include "tcp_cmd_interface.h"     // for TcpCmdInterface
00031 #include <pmt/pmt.h>
00032 #include <array>     // for array
00033 #include <cstddef>   // for size_t
00034 #include <cstdint>   // for int32_t
00035 #include <memory>    // for shared_ptr
00036 #include <string>    // for string
00037 #include <thread>    // for std::thread
00038 #include <typeinfo>  // for std::type_info, typeid
00039 #include <utility>   // for pair
00040 #include <vector>    // for vector
00041
00042 #ifdef ENABLE_FPGA
00043 #include <boost/thread.hpp>  // for boost::thread
00044 #endif
00045
00046 /** \addtogroup Core Core GNSS Receiver
00047  * Core GNSS Receiver.
00048  * \{ */
00049 /** \addtogroup Core_Receiver
00050  * Classes for the core GNSS receiver.
00051  * \{ */
00052
00053
00054 class ConfigurationInterface;
00055 class GNSSFlowgraph;
00056 class Gnss_Satellite;
00057
00058
00059 /*!
00060  * \brief This class represents the main thread of the application, so the name is ControlThread.
00061  * This is the GNSS Receiver Control Plane: it connects the flowgraph, starts running it,
00062  * and while it does not stop, reads the control messages generated by the blocks,
00063  * processes them, and applies the corresponding actions.
00064  */
00065 class ControlThread
00066 {
00067 public:
00068     static ControlThread *me;
00069     /*!
00070      * \brief Default constructor
00071      */
00072     ControlThread();
00073
00074     /*!
00075      * \brief Constructor that initializes the class with parameters
00076      *
00077      * \param[in] configuration Pointer to a ConfigurationInterface
00078      */
00079     explicit ControlThread(std::shared_ptr<ConfigurationInterface> configuration);
00080
00081     /*!
00082      * \brief Destructor
00083      */
00084     ~ControlThread();
00085
00086     /*! \brief Runs the control thread
00087      *
00088      *  This is the main loop that reads and process the control messages:
00089      *
```

```
00090      *  - Connect the GNSS receiver flowgraph;
00091      *
00092      *  - Start the GNSS receiver flowgraph;
00093      *
00094      *  while (flowgraph_->running() && !stop_){
00095      *
00096      *  - Read control messages and process them; }
00097      */
00098     int run();
00099
00100     /*!
00101      * \brief Sets the control_queue
00102      *
00103      * \param[in] std::shared_ptr<Concurrent_Queue<pmt::pmt_t» control_queue
00104      */
00105     void set_control_queue(std::shared_ptr<Concurrent_Queue<pmt::pmt_t>> control_queue);
00106
00107     unsigned int processed_control_messages() const
00108     {
00109         return processed_control_messages_;
00110     }
00111
00112     unsigned int applied_actions() const
00113     {
00114         return applied_actions_;
00115     }
00116
00117     /*!
00118      * \brief Instantiates a flowgraph
00119      *
00120      * \return Returns a smart pointer to a flowgraph object
00121      */
00122     std::shared_ptr<GNSSFlowgraph> flowgraph()
00123     {
00124         return flowgraph_;
00125     }
00126
00127 private:
00128     /*
00129      * Callback function for handling signals.
00130      * sig   identifier of signal
00131      */
00132     static void handle_signal(int sig);
00133
00134     void init();
00135
00136     void apply_action(unsigned int what);
00137
00138     /*
00139      * New receiver event dispatcher
00140      */
00141     void event_dispatcher(bool &valid_event, pmt::pmt_t &msg);
00142
00143     // Read {ephemeris, iono, utc, ref loc, ref time} assistance from a local XML file previously
      recorded
00144     bool read_assistance_from_XML();
00145
00146     /*
00147      * Blocking function that reads the GPS assistance queue
00148      */
00149     void gps_acq_assist_data_collector() const;
00150
00151     /*
00152      * Compute elevations for the specified time and position for all the available satellites in
      ephemeris and almanac queues
00153      * returns a vector filled with the available satellites ordered from high elevation to low
      elevation angle.
00154      */
00155     std::vector<std::pair<int, Gnss_Satellite» get_visible_sats(time_t rx_utc_time, const
      std::array<float, 3> &LLH);
00156
00157     /*
00158      * Read initial GNSS assistance from SUPL server or local XML files
00159      */
00160     void assist_GNSS();
00161
00162     void telecommand_listener();
00163     void keyboard_listener();
00164     void message_queue_listener();
00165     void print_help_at_exit() const;
00166
00167     // default filename for assistance data
00168     const std::string eph_default_xml_filename_ = "./gps_ephemeris.xml";
00169     const std::string utc_default_xml_filename_ = "./gps_utc_model.xml";
00170     const std::string iono_default_xml_filename_ = "./gps_iono.xml";
00171     const std::string ref_time_default_xml_filename_ = "./gps_ref_time.xml";
00172     const std::string ref_location_default_xml_filename_ = "./gps_ref_location.xml";
```

```
00173     const std::string eph_gal_default_xml_filename_ = "./gal_ephemeris.xml";
00174     const std::string eph_cnav_default_xml_filename_ = "./gps_cnav_ephemeris.xml";
00175     const std::string gal_iono_default_xml_filename_ = "./gal_iono.xml";
00176     const std::string gal_utc_default_xml_filename_ = "./gal_utc_model.xml";
00177     const std::string cnav_utc_default_xml_filename_ = "./gps_cnav_utc_model.xml";
00178     const std::string eph_glo_gnav_default_xml_filename_ = "./glo_gnav_ephemeris.xml";
00179     const std::string glo_utc_default_xml_filename_ = "./glo_utc_model.xml";
00180     const std::string gal_almanac_default_xml_filename_ = "./gal_almanac.xml";
00181     const std::string gps_almanac_default_xml_filename_ = "./gps_almanac.xml";
00182
00183     const std::string control_message_queue_name_ = "receiver_control_queue";
00184
00185     const size_t channel_event_type_hash_code_ = typeid(channel_event_sptr).hash_code();
00186     const size_t command_event_type_hash_code_ = typeid(command_event_sptr).hash_code();
00187
00188     std::shared_ptr<ConfigurationInterface> configuration_;
00189     std::shared_ptr<Concurrent_Queue<pmt::pmt_t» control_queue_;
00190     std::shared_ptr<GNSSFlowgraph> flowgraph_;
00191
00192     std::thread cmd_interface_thread_;
00193     std::thread keyboard_thread_;
00194     std::thread message_queue_thread_;
00195     std::thread gps_acq_assist_data_collector_thread_;
00196
00197 #ifdef ENABLE_FPGA
00198     boost::thread fpga_helper_thread_;
00199 #endif
00200
00201     TcpCmdInterface cmd_interface_;
00202
00203     // SUPL assistance classes
00204     Gnss_Sdr_Supl_Client supl_client_acquisition_;
00205     Gnss_Sdr_Supl_Client supl_client_ephemeris_;
00206     int supl_mcc_;  // Current network MCC (Mobile country code), 3 digits.
00207     int supl_mns_;  // Current network MNC (Mobile Network code), 2 or 3 digits.
00208     int supl_lac_;  // Current network LAC (Location area code),16 bits, 1-65520 are valid values.
00209     int supl_ci_;   // Cell Identity (16 bits, 0-65535 are valid values).
00210
00211     Agnss_Ref_Location agnss_ref_location_;
00212     Agnss_Ref_Time agnss_ref_time_;
00213
00214     unsigned int processed_control_messages_;
00215     unsigned int applied_actions_;
00216
00217     bool well_formatted_configuration_;
00218     bool conf_file_has_section_;
00219     bool conf_file_has_mandatory_globals_;
00220     bool conf_has_signal_sources_;
00221     bool conf_has_observables_;
00222     bool conf_has_pvt_;
00223     bool receiver_on_standby_;
00224     bool stop_;
00225     bool restart_;
00226     bool telecommand_enabled_;
00227 };
00228
00229
00230 /** \} */
00231 /** \} */
00232 #endif  // GNSS_SDR_CONTROL_THREAD_H
```

13.620 file_configuration.h File Reference

A ConfigurationInterface that reads the configuration from a file.

```
#include "INIReader.h"
#include "configuration_interface.h"
#include "in_memory_configuration.h"
#include "string_converter.h"
#include <cstdint>
#include <memory>
#include <string>
```

Classes

- class FileConfiguration

  This class is an implementation of the interface ConfigurationInterface.

13.620.1 Detailed Description

A ConfigurationInterface that reads the configuration from a file.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

This implementation has a text file as the source for the values of the parameters. The file is in the INI format, containing sections and pairs of names and values. For more information about the INI format, see https://en.wikipedia.org/wiki/INI_file

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file file_configuration.h.

13.621 file_configuration.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file file_configuration.h
00003  * \brief A ConfigurationInterface that reads the configuration from a file.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * This implementation has a text file as the source for the values of the parameters.
00007  * The file is in the INI format, containing sections and pairs of names and values.
00008  * For more information about the INI format, see https://en.wikipedia.org/wiki/INI_file
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021
00022 #ifndef GNSS_SDR_FILE_CONFIGURATION_H
00023 #define GNSS_SDR_FILE_CONFIGURATION_H
00024
00025 #include "INIReader.h"
00026 #include "configuration_interface.h"
00027 #include "in_memory_configuration.h"
00028 #include "string_converter.h"
00029 #include <cstdint>
00030 #include <memory>
00031 #include <string>
00032
00033 /** \addtogroup Core
00034  * \{ */
00035 /** \addtogroup Core_Receiver
00036  * \{ */
00037
00038
00039 /*!
00040  * \brief This class is an implementation of the interface ConfigurationInterface
00041  *
00042  * Derived from ConfigurationInterface, this class implements an interface
00043  * to a configuration file. This implementation has a text file as the source
00044  * for the values of the parameters.
00045  * The file is in the INI format, containing sections and pairs of names and values.
00046  * For more information about the INI format, see https://en.wikipedia.org/wiki/INI_file
00047  */
00048 class FileConfiguration : public ConfigurationInterface
00049 {
00050 public:
00051     explicit FileConfiguration(std::string filename);
00052     FileConfiguration();
00053     ~FileConfiguration() = default;
00054     std::string property(std::string property_name, std::string default_value) const override;
00055     bool property(std::string property_name, bool default_value) const override;
00056     int64_t property(std::string property_name, int64_t default_value) const override;
00057     uint64_t property(std::string property_name, uint64_t default_value) const override;
00058     int32_t property(std::string property_name, int32_t default_value) const override;
00059     uint32_t property(std::string property_name, uint32_t default_value) const override;
00060     int16_t property(std::string property_name, int16_t default_value) const override;
00061     uint16_t property(std::string property_name, uint16_t default_value) const override;
00062     float property(std::string property_name, float default_value) const override;
```

```
00063     double property(std::string property_name, double default_value) const override;
00064     void set_property(std::string property_name, std::string value) override;
00065     bool is_present(const std::string& property_name) const override;
00066     bool has_section() const;
00067
00068 private:
00069     void init();
00070     std::string filename_;
00071     std::unique_ptr<INIReader> ini_reader_;
00072     std::unique_ptr<InMemoryConfiguration> overrided_;
00073     std::unique_ptr<StringConverter> converter_;
00074     int error_{};
00075 };
00076
00077
00078 /** \} */
00079 /** \} */
00080 #endif  // GNSS_SDR_FILE_CONFIGURATION_H
```

13.622 gnss_block_factory.h File Reference

Interface of a factory that returns smart pointers to GNSS blocks.

```
#include "concurrent_queue.h"
#include <pmt/pmt.h>
#include <memory>
#include <string>
#include <vector>
```

Namespaces

- namespace block_factory

  Class that produces all kinds of GNSS blocks.

Functions

- std::unique_ptr< SignalSourceInterface > **block_factory::GetSignalSource** (const ConfigurationInterface *configuration, Concurrent_Queue< pmt::pmt_t > *queue, int ID=-1)
- std::unique_ptr< GNSSBlockInterface > **block_factory::GetSignalConditioner** (const ConfigurationInterface *configuration, int ID=-1)
- std::vector< std::unique_ptr< GNSSBlockInterface > > **block_factory::GetChannels** (const ConfigurationInterface *configuration, Concurrent_Queue< pmt::pmt_t > *queue)
- std::unique_ptr< GNSSBlockInterface > **block_factory::GetObservables** (const ConfigurationInterface *configuration)
- std::unique_ptr< GNSSBlockInterface > **block_factory::GetPVT** (const ConfigurationInterface *configuration)
- std::unique_ptr< GNSSBlockInterface > block_factory::GetBlock (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams, Concurrent_Queue< pmt::pmt_t > *queue=nullptr)

  Returns the block with the required role implementation and its configuration parameters.
- std::string block_factory::get_role_name (const ConfigurationInterface *configuration, const std::string &role_prefix, const std::string &signal, int channel)

  Returns the role name of a channel's block: role_prefix + signal + channel (e.g. "Acquisition_5X12") if that per-channel role has an .implementation property in the configuration, role_prefix + signal (e.g. "Acquisition_5X") otherwise. Use it to read a property of an already built channel block from the same role the block was instantiated with.
- std::unique_ptr< GNSSBlockInterface > **block_factory::GetChannel** (const ConfigurationInterface *configuration, const std::string &signal, int channel, Concurrent_Queue< pmt::pmt_t > *queue)
- std::unique_ptr< AcquisitionInterface > **block_factory::GetAcqBlock** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::unique_ptr< TrackingInterface > **block_factory::GetTrkBlock** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)
- std::unique_ptr< TelemetryDecoderInterface > **block_factory::GetTlmBlock** (const ConfigurationInterface *configuration, const std::string &role, unsigned int in_streams, unsigned int out_streams)

13.622.1 Detailed Description

Interface of a factory that returns smart pointers to GNSS blocks.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com Luis Esteve, 2011. luis(at)epsilon-formacion.com Javier Arribas, 2011. jarribas(at)cttc.es Carles Fernandez-Prades, 2014-2020. cfernandez(at)cttc.es

This class encapsulates the complexity behind the instantiation of GNSS blocks.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_block_factory.h.

13.623 gnss_block_factory.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_block_factory.h
00003  * \brief Interface of a factory that returns smart pointers to GNSS blocks.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *         Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *         Javier Arribas, 2011. jarribas(at)cttc.es
00007  *         Carles Fernandez-Prades, 2014-2020. cfernandez(at)cttc.es
00008  *
00009  * This class encapsulates the complexity behind the instantiation
00010  * of GNSS blocks.
00011  *
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_BLOCK_FACTORY_H
00025 #define GNSS_SDR_BLOCK_FACTORY_H
00026
00027 #include "concurrent_queue.h"
00028 #include <pmt/pmt.h>
00029 #include <memory>  // for unique_ptr
00030 #include <string>  // for string
00031 #include <vector>  // for vector
00032
00033 /** \addtogroup Core
00034  * \{ */
00035 /** \addtogroup Core_Receiver
00036  * \{ */
00037
00038
00039 class ConfigurationInterface;
00040 class GNSSBlockInterface;
00041 class SignalSourceInterface;
00042 class AcquisitionInterface;
00043 class TrackingInterface;
00044 class TelemetryDecoderInterface;
00045
00046 /*!
00047  * \brief Class that produces all kinds of GNSS blocks
00048  */
00049
00050 namespace block_factory
00051 {
00052
00053 std::unique_ptr<SignalSourceInterface> GetSignalSource(const ConfigurationInterface* configuration,
      Concurrent_Queue<pmt::pmt_t>* queue, int ID = -1);
00054
00055 std::unique_ptr<GNSSBlockInterface> GetSignalConditioner(const ConfigurationInterface* configuration,
      int ID = -1);
00056
00057 std::vector<std::unique_ptr<GNSSBlockInterface>> GetChannels(const ConfigurationInterface*
      configuration, Concurrent_Queue<pmt::pmt_t>* queue);
00058
00059 std::unique_ptr<GNSSBlockInterface> GetObservables(const ConfigurationInterface* configuration);
00060
00061 std::unique_ptr<GNSSBlockInterface> GetPVT(const ConfigurationInterface* configuration);
```

```
00062
00063 /*!
00064  * \brief Returns the block with the required role implementation and its configuration parameters
00065  */
00066 std::unique_ptr<GNSSBlockInterface> GetBlock(const ConfigurationInterface* configuration,
00067     const std::string& role,
00068     unsigned int in_streams,
00069     unsigned int out_streams,
00070     Concurrent_Queue<pmt::pmt_t>* queue = nullptr);
00071
00072 /*!
00073  * \brief Returns the role name of a channel's block: role_prefix + signal +
00074  * channel (e.g. "Acquisition_5X12") if that per-channel role has an
00075  * .implementation property in the configuration, role_prefix + signal (e.g.
00076  * "Acquisition_5X") otherwise. Use it to read a property of an already built
00077  * channel block from the same role the block was instantiated with.
00078  */
00079 std::string get_role_name(const ConfigurationInterface* configuration,
00080     const std::string& role_prefix,
00081     const std::string& signal,
00082     int channel);
00083
00084 std::unique_ptr<GNSSBlockInterface> GetChannel(
00085     const ConfigurationInterface* configuration,
00086     const std::string& signal,
00087     int channel,
00088     Concurrent_Queue<pmt::pmt_t>* queue);
00089
00090 std::unique_ptr<AcquisitionInterface> GetAcqBlock(
00091     const ConfigurationInterface* configuration,
00092     const std::string& role,
00093     unsigned int in_streams,
00094     unsigned int out_streams);
00095
00096 std::unique_ptr<TrackingInterface> GetTrkBlock(
00097     const ConfigurationInterface* configuration,
00098     const std::string& role,
00099     unsigned int in_streams,
00100     unsigned int out_streams);
00101
00102 std::unique_ptr<TelemetryDecoderInterface> GetTlmBlock(
00103     const ConfigurationInterface* configuration,
00104     const std::string& role,
00105     unsigned int in_streams,
00106     unsigned int out_streams);
00107
00108 };  // namespace block_factory
00109
00110 /** \} */
00111 /** \} */
00112 #endif  // GNSS_SDR_BLOCK_FACTORY_H
```

13.624 gnss_flowgraph.h File Reference

Interface of a GNSS receiver flow graph.

```
#include "channel_status_msg_receiver.h"
#include "concurrent_queue.h"
#include "galileo_e6_has_msg_receiver.h"
#include "galileo_tow_map.h"
#include "gnss_sdr_sample_counter.h"
#include "gnss_signal.h"
#include "osnma_msg_receiver.h"
#include "pvt_interface.h"
#include "satellite_visibility.h"
#include <gnuradio/blocks/null_sink.h>
#include <gnuradio/runtime_types.h>
#include <pmt/pmt.h>
#include <array>
#include <chrono>
#include <ctime>
#include <list>
#include <map>
#include <memory>
#include <mutex>
```

```
#include <set>
#include <string>
#include <unordered_map>
#include <utility>
#include <vector>
```

Classes

- class GNSSFlowgraph

  This class represents a GNSS flow graph.

13.624.1 Detailed Description

Interface of a GNSS receiver flow graph.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com Luis Esteve, 2011. luis(at)epsilon-formacion.com Carles Fernandez-Prades, 2014-2020. cfernandez(at)cttc.es Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.↩ com

It contains a signal source, a signal conditioner, a set of channels, an observables block and a pvt.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_flowgraph.h.

13.625 gnss_flowgraph.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_flowgraph.h
00003  * \brief Interface of a GNSS receiver flow graph.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *         Luis Esteve, 2011. luis(at)epsilon-formacion.com
00006  *         Carles Fernandez-Prades, 2014-2020. cfernandez(at)cttc.es
00007  *         Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00008  *
00009  * It contains a signal source,
00010  * a signal conditioner, a set of channels, an observables block and a pvt.
00011  *
00012  *
00013  * -----------------------------------------------------------------------------
00014  *
00015  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00016  * This file is part of GNSS-SDR.
00017  *
00018  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00019  * SPDX-License-Identifier: GPL-3.0-or-later
00020  *
00021  * -----------------------------------------------------------------------------
00022  */
00023
00024 #ifndef GNSS_SDR_GNSS_FLOWGRAPH_H
00025 #define GNSS_SDR_GNSS_FLOWGRAPH_H
00026
00027 #include "channel_status_msg_receiver.h"
00028 #include "concurrent_queue.h"
00029 #include "galileo_e6_has_msg_receiver.h"
00030 #include "galileo_tow_map.h"
00031 #include "gnss_sdr_sample_counter.h"
00032 #include "gnss_signal.h"
00033 #include "osnma_msg_receiver.h"
00034 #include "pvt_interface.h"
00035 #include "satellite_visibility.h"
00036 #include <gnuradio/blocks/null_sink.h>  // for null_sink
00037 #include <gnuradio/runtime_types.h>     // for basic_block_sptr, top_block_sptr
00038 #include <pmt/pmt.h>                    // for pmt_t
00039 #include <array>                        // for array
00040 #include <chrono>                       // for steady_clock
00041 #include <ctime>                        // for time_t
00042 #include <list>                         // for list
00043 #include <map>                          // for map
```

```
00044 #include <memory>                        // for for shared_ptr, dynamic_pointer_cast
00045 #include <mutex>                         // for mutex
00046 #include <set>                           // for set
00047 #include <string>                        // for string
00048 #include <unordered_map>                 // for unordered_map
00049 #include <utility>                       // for pair
00050 #include <vector>                        // for vector
00051 #if ENABLE_FPGA
00052 #include "gnss_sdr_fpga_sample_counter.h"
00053 #endif
00054
00055 /** \addtogroup Core
00056  * \{ */
00057 /** \addtogroup Core_Receiver
00058  * \{ */
00059
00060
00061 class ChannelInterface;
00062 class ConfigurationInterface;
00063 class GNSSBlockInterface;
00064 class Gnss_Satellite;
00065 class SignalSourceInterface;
00066
00067 /*! \brief This class represents a GNSS flow graph.
00068  *
00069  * It contains a signal source,
00070  * a signal conditioner, a set of channels, a PVT and an output filter.
00071  */
00072 class GNSSFlowgraph
00073 {
00074 public:
00075     /*!
00076      * \brief Constructor that initializes the receiver flow graph
00077      */
00078     GNSSFlowgraph(std::shared_ptr<ConfigurationInterface> configuration,
      std::shared_ptr<Concurrent_Queue<pmt::pmt_t>> queue);
00079
00080     /*!
00081      * \brief Destructor
00082      */
00083     ~GNSSFlowgraph();
00084
00085     /*!
00086      * \brief Start the flow graph
00087      */
00088     void start();
00089
00090     /*!
00091      * \brief Stop the flow graph
00092      */
00093     void stop();
00094
00095     /*!
00096      * \brief Connects the defined blocks in the flow graph
00097      *
00098      * Signal Source > Signal conditioner > Channels » Observables » PVT > Output filter
00099      */
00100     void connect();
00101
00102     /*!
00103      * \brief Disconnect the blocks in the flow graph
00104      */
00105     void disconnect();
00106
00107     /*!
00108      * \brief Wait for a flowgraph to complete.
00109      *
00110      * Flowgraphs complete when either
00111      * (1) all blocks indicate that they are done, or
00112      * (2) after stop() has been called to request shutdown.
00113      */
00114     void wait();
00115
00116     /*!
00117      * \brief Manage satellite acquisition
00118      *
00119      * \param[in] who   Channel ID
00120      */
00121     void acquisition_manager(unsigned int who);
00122
00123     /*!
00124      * \brief Applies an action to the flow graph
00125      *
00126      * \param[in] who   Who generated the action
00127      * \param[in] what  What is the action. 0: acquisition failed; 1: acquisition success; 2: tracking
      lost
00128      */
```

```
00129     void apply_action(unsigned int who, unsigned int what);
00130
00131     /*!
00132      * \brief Set flow graph configuratiob
00133      */
00134     void set_configuration(const std::shared_ptr<ConfigurationInterface>& configuration);
00135
00136     bool connected() const
00137     {
00138         return connected_;
00139     }
00140
00141     bool running() const
00142     {
00143         return running_;
00144     }
00145
00146     /*!
00147      * \brief Sends a GNU Radio asynchronous message from telemetry to PVT
00148      *
00149      * It is used to assist the receiver with external ephemeris data
00150      */
00151     bool send_telemetry_msg(const pmt::pmt_t& msg);
00152
00153     /*!
00154      * \brief Returns a smart pointer to the PVT object
00155      */
00156     std::shared_ptr<PvtInterface> get_pvt()
00157     {
00158         return std::dynamic_pointer_cast<PvtInterface>(pvt_);
00159     }
00160
00161     /*!
00162      * \brief Prioritize visible satellites in the specified vector
00163      */
00164     void priorize_satellites(const std::vector<std::pair<int, Gnss_Satellite»& visible_satellites);
00165
00166     /*!
00167      * \brief Re-evaluates satellite visibility when warranted (new fix, new
00168      * ephemeris/almanac, or recompute interval elapsed) so that subsequent
00169      * search_next_signal() calls favor visible satellites.
00170      * No-op unless GNSS-SDR.enable_visibility_aware_search=true.
00171      */
00172     void MaybeUpdateVisibility();
00173
00174     /*!
00175      * \brief Sets the telecommand position/time reference and immediately
00176      * refreshes visibility. Called by the control thread.
00177      */
00178     void UpdateVisibilityReference(time_t utc_time, const std::array<float, 3>& LLH);
00179
00180     /*!
00181      * \brief Whether GNSS-SDR.enable_visibility_aware_search is on, so callers can
00182      * skip the legacy get_visible_sats()/priorize_satellites() startup reorder that
00183      * MaybeUpdateVisibility() supersedes.
00184      */
00185     bool visibility_aware_search_enabled() const;
00186
00187     /*!
00188      * \brief Stops any channel whose decoded satellite is already tracked, on the
00189      * same signal, by another channel with higher C/N0. The stopped channel's
00190      * assignment returns to the search pool and the channel is re-dispatched.
00191      *
00192      * GLONASS channels are assigned a frequency channel and only learn their
00193      * orbital slot from the navigation message, so a false lock can be relabelled
00194      * as a satellite another channel already tracks; duplicate observations of
00195      * one satellite break the PVT solution.
00196      */
00197     void stop_duplicated_satellite_channels();
00198
00199 #if ENABLE_FPGA
00200     void start_acquisition_helper();
00201
00202     void perform_hw_reset();
00203 #endif
00204
00205 private:
00206     uint32_t doppler_bins_for_uncertainty(const std::string& acq_role, double uncertainty_hz) const;
00207     void init();  // Populates the SV PRN list available for acquisition and tracking
00208     int connect_desktop_flowgraph();
00209
00210     int connect_signal_sources();
00211     int connect_signal_conditioners();
00212     int connect_channels();
00213     int connect_observables();
00214     int connect_pvt();
00215     int connect_sample_counter();
```

```
00216     int connect_galileo_tow_map();
00217
00218     int connect_signal_sources_to_signal_conditioners();
00219     int connect_signal_conditioners_to_channels();
00220     int connect_channels_to_observables();
00221     int connect_observables_to_pvt();
00222     int connect_monitors();
00223     int connect_osnma();
00224     int connect_gal_e6_has();
00225     int connect_gnss_synchro_monitor();
00226     int connect_acquisition_monitor();
00227     int connect_tracking_monitor();
00228     int connect_navdata_monitor();
00229
00230 #if ENABLE_FPGA
00231     int connect_fpga_flowgraph();
00232     int connect_fpga_sample_counter();
00233 #endif
00234
00235     int assign_channels();
00236     void check_signal_conditioners();
00237
00238     void set_signals_list();
00239     void keep_one_glonass_slot_per_frequency(std::set<unsigned int>& available_prns);
00240
00241     void set_channels_state();  // Initializes the channels state (start acquisition or keep standby)
00242                                 // using the configuration parameters (number of channels and max
      channels in acquisition)
00243     //! On assistance, estimated_doppler is returned already projected to the
00244     //! searched signal's carrier frequency (see project_doppler()).
00245     Gnss_Signal search_next_signal(const std::string& searched_signal,
00246         bool& is_primary_frequency,
00247         bool& assistance_available,
00248         float& estimated_doppler,
00249         double& RX_time,
00250         bool& signal_available);
00251
00252     void push_back_signal(const Gnss_Signal& gs);
00253     void remove_signal(const Gnss_Signal& gs);
00254
00255     // Pop by visibility ratio, preserving FIFO order within each bucket and
00256     // falling back to the other bucket if empty. Keep excluded entries queued;
00257     // set picked=false when nothing is searchable.
00258     Gnss_Signal pop_by_visibility(std::list<Gnss_Signal>& available_signals, const std::string&
      searched_signal, bool& picked, double cooldown_receiver_time_s);
00259     void print_help();
00260     void check_desktop_conf_in_fpga_env();
00261
00262     double project_doppler(const std::string& searched_signal, const std::string& assist_signal,
      double assist_doppler_hz);
00263     bool is_multiband() const;
00264
00265     std::vector<std::string> split_string(const std::string& s, char delim);
00266     std::vector<bool> signal_conditioner_connected_;
00267
00268     gr::top_block_sptr top_block_;
00269
00270     std::shared_ptr<ConfigurationInterface> configuration_;
00271     std::shared_ptr<Concurrent_Queue<pmt::pmt_t» queue_;
00272
00273     std::vector<std::shared_ptr<SignalSourceInterface» sig_source_;
00274     std::vector<std::shared_ptr<GNSSBlockInterface» sig_conditioner_;
00275     std::vector<std::shared_ptr<ChannelInterface» channels_;
00276     std::shared_ptr<GNSSBlockInterface> observables_;
00277     std::shared_ptr<GNSSBlockInterface> pvt_;
00278
00279     std::map<std::string, gr::basic_block_sptr> acq_resamplers_;
00280     std::vector<gr::blocks::null_sink::sptr> null_sinks_;
00281
00282     gr::basic_block_sptr GnssSynchroMonitor_;
00283     gr::basic_block_sptr GnssSynchroAcquisitionMonitor_;
00284     gr::basic_block_sptr GnssSynchroTrackingMonitor_;
00285     gr::basic_block_sptr NavDataMonitor_;
00286     channel_status_msg_receiver_sptr channels_status_;  // class that receives and stores the current
      status of the receiver channels
00287     galileo_e6_has_msg_receiver_sptr gal_e6_has_rx_;
00288     galileo_tow_map_sptr galileo_tow_map_;
00289     osnma_msg_receiver_sptr osnma_rx_;
00290
00291     gnss_sdr_sample_counter_sptr ch_out_sample_counter_;
00292 #if ENABLE_FPGA
00293     gnss_sdr_fpga_sample_counter_sptr ch_out_fpga_sample_counter_;
00294 #endif
00295
00296     std::vector<unsigned int> channels_state_;  // 0 - Idle, 1 - Assigned, 2 - Acquisition, 3 -
      Tracking
00297
```

```
00298     std::unordered_map<std::string, std::list<Gnss_Signal» available_signals_map_;
00299
00300     std::unique_ptr<SatelliteVisibility> satellite_visibility_;
00301     std::unordered_map<std::string, uint32_t> visibility_pick_counter_;  // per signal_str,
      ratio-based visible/maybe-visible cycling
00302     // Cache exhausted pools to avoid repeated scans and shared status-map copies.
00303     // Invalidate on push_back_signal() or visibility changes.
00304     std::set<std::string> signals_with_nothing_searchable_;
00305     // Rate-limits the "no assist-tracked satellite available" log in
00306     // search_next_signal(): the condition is re-evaluated for every idle channel at
00307     // ~10 Hz and would otherwise flood the log.
00308     std::unordered_map<std::string, std::chrono::steady_clock::time_point> no_assist_log_throttle_;
00309
00310     // Last acquisition attempt per PRN/signal, in sample time for replay support.
00311     // GNSS-SDR.acquisition_max_retry_rate_hz sets the limit (0 disables it).
00312     double acquisition_retry_min_interval_s_;
00313     std::unordered_map<std::string, double> last_acquisition_attempt_rx_time_s_;
00314     bool InAcquisitionCooldown(const Gnss_Signal& gs, double receiver_time_s) const;
00315     void MarkAcquisitionAttempt(const Gnss_Signal& gs, double receiver_time_s);
00316
00317     enum StringValue
00318     {
00319         evGPS_1C,
00320         evGPS_2S,
00321         evGPS_L5,
00322         evSBAS_1C,
00323         evGAL_1B,
00324         evGAL_5X,
00325         evGAL_7X,
00326         evGAL_E6,
00327         evGLO_1G,
00328         evGLO_2G,
00329         evBDS_B1,
00330         evBDS_B1C,
00331         evBDS_B2A,
00332         evBDS_B3,
00333         evQZS_J1,
00334         evQZS_J5
00335     };
00336     std::map<std::string, StringValue> mapStringValues_;
00337
00338     std::string config_file_;
00339     std::string help_hint_;
00340
00341     std::mutex signal_list_mutex_;
00342
00343     int sources_count_;
00344     int channels_count_;
00345     int acq_channels_count_;
00346     int max_acq_channels_;
00347
00348     bool connected_;
00349     bool running_;
00350     bool multiband_;
00351     bool enable_monitor_;
00352     bool enable_acquisition_monitor_;
00353     bool enable_tracking_monitor_;
00354     bool enable_navdata_monitor_;
00355     bool enable_fpga_offloading_;
00356     bool enable_osnma_rx_;
00357     bool enable_e6_has_rx_;
00358     bool enable_secondary_signal_status_gating_;
00359 };
00360
00361
00362 /** \} */
00363 /** \} */
00364 #endif  // GNSS_SDR_GNSS_FLOWGRAPH_H
```

13.626 in_memory_configuration.h File Reference

A ConfigurationInterface for testing purposes.

```
#include "configuration_interface.h"
#include "string_converter.h"
#include <cstdint>
#include <map>
#include <memory>
#include <string>
```

Classes

- class InMemoryConfiguration

  This class is an implementation of the interface ConfigurationInterface.

13.626.1 Detailed Description

A ConfigurationInterface for testing purposes.

Author

Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com

This implementation accepts configuration parameters upon instantiation and it is intended to be used in unit testing.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file in_memory_configuration.h.

13.627 in_memory_configuration.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file in_memory_configuration.h
00003  * \brief  A ConfigurationInterface for testing purposes.
00004  * \author Carlos Aviles, 2010. carlos.avilesr(at)googlemail.com
00005  *
00006  * This implementation accepts configuration parameters upon instantiation and
00007  * it is intended to be used in unit testing.
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020
00021 #ifndef GNSS_SDR_IN_MEMORY_CONFIGURATION_H
00022 #define GNSS_SDR_IN_MEMORY_CONFIGURATION_H
00023
00024 #include "configuration_interface.h"
00025 #include "string_converter.h"
00026 #include <cstdint>
00027 #include <map>
00028 #include <memory>
00029 #include <string>
00030
00031 /** \addtogroup Core
00032  * \{ */
00033 /** \addtogroup Core_Receiver
00034  * \{ */
00035
00036
00037 /*!
00038  * \brief  This class is an implementation of the interface ConfigurationInterface.
00039  *
00040  * This implementation accepts configuration parameters upon instantiation and
00041  * it is intended to be used in unit testing.
00042  */
00043 class InMemoryConfiguration : public ConfigurationInterface
00044 {
00045 public:
00046     InMemoryConfiguration();
00047     ~InMemoryConfiguration();
00048     std::string property(std::string property_name, std::string default_value) const override;
00049     bool property(std::string property_name, bool default_value) const override;
00050     int64_t property(std::string property_name, int64_t default_value) const override;
00051     uint64_t property(std::string property_name, uint64_t default_value) const override;
00052     int32_t property(std::string property_name, int32_t default_value) const override;
00053     uint32_t property(std::string property_name, uint32_t default_value) const override;
00054     int16_t property(std::string property_name, int16_t default_value) const override;
00055     uint16_t property(std::string property_name, uint16_t default_value) const override;
00056     float property(std::string property_name, float default_value) const override;
00057     double property(std::string property_name, double default_value) const override;
```

```
00058     void set_property(std::string property_name, std::string value) override;
00059     void supersede_property(const std::string& property_name, const std::string& value);
00060     bool is_present(const std::string& property_name) const override;
00061
00062 private:
00063     std::map<std::string, std::string> properties_;
00064     std::unique_ptr<StringConverter> converter_;
00065 };
00066
00067
00068 /** \} */
00069 /** \} */
00070 #endif  // GNSS_SDR_IN_MEMORY_CONFIGURATION_H
```

13.628 satellite_visibility.h File Reference

Runtime satellite-visibility tracking for acquisition search prioritization.

```
#include "gnss_satellite.h"
#include "rtklib.h"
#include <armadillo>
#include <array>
#include <cstdint>
#include <ctime>
#include <limits>
#include <map>
#include <memory>
#include <set>
#include <string>
#include <tuple>
#include <utility>
#include <vector>
```

Classes

- class SatelliteVisibility

  Opt-in acquisition search classification: visible, excluded, or unknown. Healthy satellites above the search mask are preferred; unknowns are searched less often. Banned PRNs are omitted, and tracking can override exclusion. Tick() refreshes on reference, fix, navigation-data, position, interval, or expiry changes. Expired data returns satellites to the unknown pool. Disabled by default (GNSS-SDR.enable_visibility_aware_search).

Functions

- std::vector< std::pair< int, Gnss_Satellite > > compute_visible_satellites (const std::shared_ptr< PvtInterface > &pvt_ptr, const gtime_t &gps_gtime, const arma::vec &r_eb_e, double elevation_mask_deg, std::vector< std::pair< int, Gnss_Satellite > > *below_mask_out=nullptr, double almanac_max_age_s=std::numeric_limits< double >::infinity(), double *seconds_until_next_expiry_out=nullptr, const std::set< std::pair< std::string, uint32_t > > *only_prns=nullptr, bool glonass_strict_health=true)

  Return healthy GPS/Galileo/BeiDou/GLONASS/QZSS satellites above the mask. Prefer fresh ephemeris over almanac. BeiDou selects the closest orbit (ties: DNAV, CNAV1, CNAV2); B-CNAV supports IGSO/MEO only. Unhealthy satellites are excluded, not unknown; Galileo uses E1B health.

13.628.1 Detailed Description

Runtime satellite-visibility tracking for acquisition search prioritization.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file satellite_visibility.h.

13.629 satellite_visibility.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file satellite_visibility.h
00003  * \brief Runtime satellite-visibility tracking for acquisition search
00004  * prioritization.
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * SPDX-FileCopyrightText: 2026 (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SATELLITE_VISIBILITY_H
00018 #define GNSS_SDR_SATELLITE_VISIBILITY_H
00019
00020 #include "gnss_satellite.h"
00021 #include "rtklib.h"  // for gtime_t
00022 #include <armadillo>
00023 #include <array>
00024 #include <cstdint>
00025 #include <ctime>
00026 #include <limits>
00027 #include <map>
00028 #include <memory>
00029 #include <set>
00030 #include <string>
00031 #include <tuple>
00032 #include <utility>
00033 #include <vector>
00034
00035 class ConfigurationInterface;
00036 class Monitor_Pvt;
00037 class PvtInterface;
00038
00039 /** \addtogroup Core
00040  * \{ */
00041 /** \addtogroup Core_Receiver
00042  * \{ */
00043
00044 /*!
00045  * \brief Return healthy GPS/Galileo/BeiDou/GLONASS/QZSS satellites above the mask.
00046  * Prefer fresh ephemeris over almanac. BeiDou selects the closest orbit
00047  * (ties: DNAV, CNAV1, CNAV2); B-CNAV supports IGSO/MEO only.
00048  * Unhealthy satellites are excluded, not unknown; Galileo uses E1B health.
00049  *
00050  * \param gps_gtime GPST query epoch; also resolves truncated navigation weeks.
00051  * \param r_eb_e receiver ECEF position (m).
00052  * \param elevation_mask_deg search mask (deg), independent of PVT.elevation_mask.
00053  * \param below_mask_out optional output for satellites at/below the mask or unhealthy.
00054  * \param almanac_max_age_s maximum absolute almanac age (s); infinity disables it.
00055  * Ephemeris age limits use RTKLIB's per-system MAXDTOE* constants.
00056  * \param seconds_until_next_expiry_out optional time to earliest classified-data expiry;
00057  * infinity if nothing was classified.
00058  * \param only_prns optional (system, PRN) filter; requires unchanged position/time.
00059  * \param glonass_strict_health also reject the Bn MSB; ln always marks unhealthy.
00060  * \return (floor(elevation) in degrees, satellite) pairs, highest elevation first.
00061  */
00062 std::vector<std::pair<int, Gnss_Satellite» compute_visible_satellites(
00063     const std::shared_ptr<PvtInterface>& pvt_ptr,
00064     const gtime_t& gps_gtime,
00065     const arma::vec& r_eb_e,
00066     double elevation_mask_deg,
00067     std::vector<std::pair<int, Gnss_Satellite»* below_mask_out = nullptr,
00068     double almanac_max_age_s = std::numeric_limits<double>::infinity(),
00069     double* seconds_until_next_expiry_out = nullptr,
00070     const std::set<std::pair<std::string, uint32_t»* only_prns = nullptr,
00071     bool glonass_strict_health = true);
00072
00073
00074 /*!
00075  * \brief Opt-in acquisition search classification: visible, excluded, or unknown.
00076  * Healthy satellites above the search mask are preferred; unknowns are searched
00077  * less often. Banned PRNs are omitted, and tracking can override exclusion.
00078  * Tick() refreshes on reference, fix, navigation-data, position, interval, or
00079  * expiry changes. Expired data returns satellites to the unknown pool.
00080  * Disabled by default (GNSS-SDR.enable_visibility_aware_search).
00081  */
00082 class SatelliteVisibility
00083 {
```

```
00084 public:
00085     explicit SatelliteVisibility(const std::shared_ptr<ConfigurationInterface>& configuration);
00086
00087     bool enabled() const { return enabled_; }
00088     uint32_t search_ratio() const { return search_ratio_; }
00089
00090     /*!
00091      * \brief Supplies a telecommand position and UTC epoch, anchored to the
00092      * current sample clock. Forces the next Tick() to recompute and overrides
00093      * the current fix/configured reference until a different PVT epoch arrives.
00094      */
00095     void SetCommandReference(time_t utc_time, const std::array<float, 3>& LLH,
00096         const Monitor_Pvt& current_fix, double receiver_time_s);
00097
00098     /*!
00099      * \brief Refresh visibility using telecommand, fix, then AGNSS reference priority.
00100      * \param receiver_time_s elapsed sample time; advances epochs between fixes.
00101      * \returns true when classification changes require rebuilding search queues.
00102      */
00103     bool Tick(const std::shared_ptr<PvtInterface>& pvt_ptr, const Monitor_Pvt& fix_status,
00104         double receiver_time_s = 0.0);
00105
00106     bool IsVisible(const Gnss_Satellite& sat) const;
00107     // Elevation computable and at/below the mask (or unhealthy). Not the
00108     // negation of IsVisible(): a satellite with no data is in neither set.
00109     bool IsExcluded(const Gnss_Satellite& sat) const;
00110
00111     // GLONASS acquisition searches an FDMA frequency shared by orbital slots:
00112     // visible if any slot is visible, excluded only if every slot is excluded.
00113     // Other constellations retain their per-satellite classification.
00114     bool IsSearchVisible(const Gnss_Satellite& sat) const;
00115     bool IsSearchExcluded(const Gnss_Satellite& sat) const;
00116
00117     // Predict Doppler (Hz) from usable navigation data and receiver clock drift.
00118     // Live fixes must be anchored by Tick() within one second of sample time.
00119     // Pre-fix prediction requires opt-in, an AGNSS reference, and explicit bounds;
00120     // it assumes zero velocity and GNSS-SDR.clock_frequency_offset_ppm.
00121     // GLONASS requires exactly one visible slot on the searched FDMA frequency.
00122     // Returns false without changing doppler_hz if prediction is unavailable.
00123     bool PredictedDopplerHz(const std::shared_ptr<PvtInterface>& pvt_ptr,
00124         const Monitor_Pvt& fix_status, double receiver_time_s, const Gnss_Satellite& sat, const
      std::string& signal,
00125         double& doppler_hz) const;
00126
00127     // Doppler uncertainty half-width (Hz): zero for a live fix, otherwise the
00128     // configured clock-error and speed bounds projected onto this carrier.
00129     // Missing/invalid bounds return infinity; explicit zero bounds are allowed.
00130     double PredictedDopplerUncertaintyHz(const Monitor_Pvt& fix_status, const std::string& signal)
      const;
00131
00132 private:
00133     enum class SearchVisibility
00134     {
00135         Unknown,
00136         Visible,
00137         Excluded
00138     };
00139
00140     SearchVisibility GetSearchVisibility(const Gnss_Satellite& sat) const;
00141
00142     // Predict for the sole visible slot on prn's FDMA frequency using ephemeris
00143     // or almanac. Returns its carrier frequency; band is 1 (L1) or 2 (L2).
00144     bool GlonassGeometricDopplerHz(const std::shared_ptr<PvtInterface>& pvt_ptr,
00145         const gtime_t& gps_gtime, uint32_t prn, int band, const std::array<double, 3>& rx_pos_m,
00146         const std::array<double, 3>& rx_vel_mps, double& geometric_doppler_hz, double&
      carrier_freq_hz) const;
00147
00148     // QZSS geometric Doppler from LNAV, CNAV, or almanac, in that order.
00149     // Maps L1 C/B PRNs to their nominal PRNs.
00150     bool QzssGeometricDopplerHz(const std::shared_ptr<PvtInterface>& pvt_ptr,
00151         const gtime_t& gps_gtime, uint32_t prn, const std::array<double, 3>& rx_pos_m,
00152         const std::array<double, 3>& rx_vel_mps, double carrier_freq_hz, double& geometric_doppler_hz)
      const;
00153
00154     // BeiDou part of PredictedDopplerHz(): use the same validated DNAV,
00155     // CNAV1, CNAV2 or fallback almanac orbit as visibility, evaluated on
00156     // the requested carrier at the receiver's ECEF position and velocity.
00157     bool BeidouGeometricDopplerHz(const std::shared_ptr<PvtInterface>& pvt_ptr,
00158         const gtime_t& gps_gtime, uint32_t prn, const std::array<double, 3>& rx_pos_m,
00159         const std::array<double, 3>& rx_vel_mps, double carrier_freq_hz, double& geometric_doppler_hz)
      const;
00160
00161     // Reports added, changed, or removed PRNs for targeted recomputation.
00162     // The first call leaves changed_prns_out untouched and forces a full sweep.
00163     bool DataChanged(const std::shared_ptr<PvtInterface>& pvt_ptr,
00164         std::set<std::pair<std::string, uint32_t>>* changed_prns_out = nullptr);
00165
```

```
00166     bool enabled_;
00167     uint32_t search_ratio_;
00168     double recompute_interval_s_;
00169     double elevation_mask_deg_;
00170
00171     // GNSS-SDR.visibility_recompute_position_threshold_m: ECEF displacement
00172     // (m) since the last full recompute that forces a new one, independently
00173     // of recompute_interval_s_.
00174     double position_threshold_m_;
00175
00176     // GNSS-SDR.visibility_almanac_max_age_s (s). Ephemeris staleness is not
00177     // configurable: RTKLIB's per-system MAXDTOE* constants apply.
00178     double almanac_max_age_s_;
00179
00180     bool glonass_strict_health_;
00181     bool have_agnss_reference_;
00182     double agnss_ref_lat_deg_;
00183     double agnss_ref_lon_deg_;
00184
00185     // GNSS-SDR.AGNSS_ref_utc_time, falling back to wall-clock time if unset.
00186     // Initial epoch for the no-fix fallback; in replay runs the wall clock is
00187     // unrelated to the GNSS time in the samples, so configure it explicitly.
00188     time_t agnss_ref_utc_time_;
00189
00190     // Opt-in pre-fix prediction from the AGNSS reference (default false).
00191     bool doppler_prediction_before_fix_;
00192
00193     // Pre-fix clock offset/error (ppm) and speed bound (m/s), from the matching
00194     // GNSS-SDR configuration keys. Live fixes supply their own clock and velocity.
00195     double clock_frequency_offset_ppm_;
00196     double clock_frequency_max_error_ppm_;
00197     double receiver_max_velocity_m_s_;
00198     bool have_doppler_uncertainty_budget_;
00199
00200     bool have_command_reference_{false};
00201     std::array<float, 3> command_reference_llh_{};
00202     time_t command_reference_utc_time_{0};
00203     double command_reference_receiver_time_s_{0.0};
00204     double command_previous_fix_time_s_{-1.0};
00205     bool command_reference_changed_{false};
00206
00207     // GNSS-SDR.<System>_banned_prns, parsed as in GNSSFlowgraph::set_signals_list().
00208     // A banned PRN is never classified visible, keeping the diagnostics
00209     // consistent with its removal from the search pool.
00210     std::set<std::pair<std::string, uint32_t» banned_;
00211
00212     // Systems with at least one Channels_<signal>.count > 0. Scopes the
00213     // maybe-visible report in Tick() so an unconfigured system's PRN range
00214     // is not listed as permanently unclassified.
00215     std::set<std::string> configured_systems_;
00216
00217     std::set<std::pair<std::string, uint32_t» visible_;
00218     std::set<std::pair<std::string, uint32_t» excluded_;  // elevation computable, at/below the mask
00219
00220     // Floored elevations for diagnostics, including unchanged PRNs during
00221     // targeted recomputes. Classification uses the computation's full-precision
00222     // mask and health decisions, never these rounded values.
00223     std::map<std::pair<std::string, uint32_t>, double> reference_elevation_deg_;
00224
00225     // Anchor the last successful fix to the elapsed sample clock. Repeated
00226     // status snapshots must not reset this anchor during a solution outage.
00227     double last_fix_time_s_{-1.0};
00228     double last_fix_receiver_time_s_{0.0};
00229
00230     // Per (system, navigation family, PRN): absolute toe/toa, health, IODE, IODC,
00231     // satellite type, and signal type. Unused legacy fields are zero.
00232     using NavigationFingerprint = std::tuple<double, int32_t, uint32_t, uint32_t, int32_t, int32_t>;
00233     std::map<std::tuple<std::string, std::string, uint32_t>, NavigationFingerprint>
      last_data_fingerprints_;
00234
00235     // Absolute GPST seconds (gtime_t.time + sec) of the last full sweep:
00236     // monotonic across week rollover, and receiver time keeps the interval
00237     // independent of replay speed.
00238     double last_recompute_rx_time_s_;
00239
00240     // ECEF position (m) of the last full sweep. Seeded at the origin so the
00241     // first displacement check trivially exceeds position_threshold_m_.
00242     arma::vec last_recompute_r_eb_e_;
00243
00244     // Receiver time (same scale as last_recompute_rx_time_s_) at which the
00245     // freshest classified data goes stale. +infinity until a full sweep with
00246     // data has run; only full sweeps update it.
00247     double next_expiry_deadline_rx_time_;
00248
00249     // Throttle navigation-map copies by tick count; check on the first Tick().
00250     int ticks_since_data_check_;
00251
```

```
00252     bool last_fix_valid_;
00253     bool first_data_check_;
00254 };
00255
00256 /** \} */
00257 /** \} */
00258 #endif  // GNSS_SDR_SATELLITE_VISIBILITY_H
```

13.630 tcp_cmd_interface.h File Reference

Class that implements a TCP/IP telecommand command line interface for GNSS-SDR.

```
#include "concurrent_queue.h"
#include <pmt/pmt.h>
#include <array>
#include <cstdint>
#include <ctime>
#include <functional>
#include <memory>
#include <string>
#include <unordered_map>
#include <vector>
```

Classes

- class TcpCmdInterface

13.630.1 Detailed Description

Class that implements a TCP/IP telecommand command line interface for GNSS-SDR.

Author

Javier Arribas jarribas (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file tcp_cmd_interface.h.

13.631 tcp_cmd_interface.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tcp_cmd_interface.h
00003  *
00004  * \brief Class that implements a TCP/IP telecommand command line interface
00005  * for GNSS-SDR
00006  * \author Javier Arribas jarribas (at) cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018 #ifndef GNSS_SDR_TCP_CMD_INTERFACE_H
00019 #define GNSS_SDR_TCP_CMD_INTERFACE_H
00020
00021
00022 #include "concurrent_queue.h"
00023 #include <pmt/pmt.h>
00024 #include <array>
00025 #include <cstdint>
00026 #include <ctime>
00027 #include <functional>
00028 #include <memory>
```

```
00029 #include <string>
00030 #include <unordered_map>
00031 #include <vector>
00032
00033 /** \addtogroup Core
00034  * \{ */
00035 /** \addtogroup Core_Receiver
00036  * \{ */
00037
00038
00039 class PvtInterface;
00040
00041 class TcpCmdInterface
00042 {
00043 public:
00044     TcpCmdInterface();
00045     ~TcpCmdInterface() = default;
00046     void run_cmd_server(int tcp_port);
00047     void set_msg_queue(std::shared_ptr<Concurrent_Queue<pmt::pmt_t>> control_queue);
00048
00049     /*!
00050      * \brief gets the UTC time parsed from the last TC command issued
00051      */
00052     time_t get_utc_time() const;
00053
00054     /*!
00055      * \brief gets the Latitude, Longitude and Altitude vector from the last TC command issued
00056      */
00057     std::array<float, 3> get_LLH() const;
00058
00059     void set_pvt(std::shared_ptr<PvtInterface> PVT_sptr);
00060
00061 private:
00062     std::unordered_map<std::string, std::function<std::string(const std::vector<std::string> &)»
00063         functions_;
00064     std::string status(const std::vector<std::string> &commandLine);
00065     std::string reset(const std::vector<std::string> &commandLine);
00066     std::string standby(const std::vector<std::string> &commandLine);
00067     std::string hotstart(const std::vector<std::string> &commandLine);
00068     std::string warmstart(const std::vector<std::string> &commandLine);
00069     std::string coldstart(const std::vector<std::string> &commandLine);
00070     std::string set_ch_satellite(const std::vector<std::string> &commandLine);
00071
00072     void register_functions();
00073
00074     std::shared_ptr<Concurrent_Queue<pmt::pmt_t» control_queue_;
00075     std::shared_ptr<PvtInterface> PVT_sptr_;
00076
00077     float rx_latitude_;
00078     float rx_longitude_;
00079     float rx_altitude_;
00080
00081     time_t receiver_utc_time_;
00082
00083     bool keep_running_;
00084 };
00085
00086
00087 /** \} */
00088 /** \} */
00089 #endif  // GNSS_SDR_TCP_CMD_INTERFACE_H
```

13.632 agnss_ref_location.h File Reference

Interface of an Assisted GNSS REFERENCE LOCATION storage.

```
#include <boost/serialization/nvp.hpp>
#include <sstream>
#include <string>
#include <vector>
```

Classes

- class Agnss_Ref_Location

  Interface of an Assisted GNSS REFERENCE LOCATION storage.

Functions

- Agnss_Ref_Location parse_agnss_ref_location (const std::string &ref_location_str)

  Parses GNSS-SDR.AGNSS_ref_location ("lat,lon" or "lat lon", degrees). valid is false if the string is empty or not a plausible WGS84 position. Single parsing point for ControlThread and SatelliteVisibility.

13.632.1 Detailed Description

Interface of an Assisted GNSS REFERENCE LOCATION storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file agnss_ref_location.h.

13.633 agnss_ref_location.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file agnss_ref_location.h
00003  * \brief  Interface of an Assisted GNSS REFERENCE LOCATION storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_AGNSS_REF_LOCATION_H
00019 #define GNSS_SDR_AGNSS_REF_LOCATION_H
00020
00021 #include <boost/serialization/nvp.hpp>
00022 #include <sstream>
00023 #include <string>
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * Classes containing info about system parameters for the different GNSS.
00030  * \{ */
00031
00032 /*!
00033  * \brief  Interface of an Assisted GNSS REFERENCE LOCATION storage
00034  *
00035  */
00036 class Agnss_Ref_Location
00037 {
00038 public:
00039     /*!
00040      * Default constructor
00041      */
00042     Agnss_Ref_Location() = default;
00043
00044     double lat{};
00045     double lon{};
00046     double uncertainty{};
00047     bool valid{};
00048
00049     template <class Archive>
00050
00051     /*!
00052      * \brief Serialize is a boost standard method to be called by the boost XML
00053      * serialization. Here is used to save the Ref location on disk file.
00054      */
00055     inline void serialize(Archive& archive, const unsigned int version)
00056     {
00057         using boost::serialization::make_nvp;
```

```
00058         if (version)
00059             {
00060             };
00061         archive& BOOST_SERIALIZATION_NVP(lat);
00062         archive& BOOST_SERIALIZATION_NVP(lon);
00063         archive& BOOST_SERIALIZATION_NVP(uncertainty);
00064         archive& BOOST_SERIALIZATION_NVP(valid);
00065     }
00066 };
00067
00068
00069 /*!
00070  * \brief Parses GNSS-SDR.AGNSS_ref_location ("lat,lon" or "lat lon", degrees).
00071  * valid is false if the string is empty or not a plausible WGS84 position.
00072  * Single parsing point for ControlThread and SatelliteVisibility.
00073  */
00074 inline Agnss_Ref_Location parse_agnss_ref_location(const std::string& ref_location_str)
00075 {
00076     Agnss_Ref_Location result{};
00077     if (ref_location_str.empty())
00078         {
00079             return result;
00080         }
00081     std::vector<double> vect;
00082     std::stringstream ss(ref_location_str);
00083     double d;
00084     while (ss » d)
00085         {
00086             vect.push_back(d);
00087             if ((ss.peek() == ',') || (ss.peek() == ' '))
00088                 {
00089                     ss.ignore();
00090                 }
00091         }
00092     if (vect.size() >= 2)
00093         {
00094             if ((vect[0] < 90.0) && (vect[0] > -90) && (vect[1] < 180.0) && (vect[1] > -180.0))
00095                 {
00096                     result.lat = vect[0];
00097                     result.lon = vect[1];
00098                     result.valid = true;
00099                 }
00100         }
00101     return result;
00102 }
00103
00104
00105 /** \} */
00106 /** \} */
00107 #endif  // GNSS_SDR_AGNSS_REF_LOCATION_H
```

13.634 agnss_ref_time.h File Reference

Interface of an Assisted GNSS REFERENCE TIME storage.

```
#include <boost/serialization/nvp.hpp>
#include <ctime>
#include <string>
```

Classes

- class Agnss_Ref_Time

  Interface of an Assisted GNSS REFERENCE TIME storage.

Functions

- Agnss_Ref_Time parse_agnss_ref_utc_time (const std::string &ref_time_str, bool *malformed_year=nullptr, bool *malformed_format=nullptr)

  Parses GNSS-SDR.AGNSS_ref_utc_time ("DD/MM/YYYY HH:MM:SS", UTC). An empty string is not an error: the host's current time is returned, marked valid. malformed_year and malformed_format (both optional) tell why a non-empty string was rejected. Single parsing point for ControlThread and SatelliteVisibility.

13.634.1 Detailed Description

Interface of an Assisted GNSS REFERENCE TIME storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file agnss_ref_time.h.

13.635 agnss_ref_time.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file agnss_ref_time.h
00003  * \brief  Interface of an Assisted GNSS REFERENCE TIME storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_AGNSS_REF_TIME_H
00019 #define GNSS_SDR_AGNSS_REF_TIME_H
00020
00021 #include <boost/serialization/nvp.hpp>
00022 #include <ctime>
00023 #include <string>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief  Interface of an Assisted GNSS REFERENCE TIME storage
00033  *
00034  */
00035 class Agnss_Ref_Time
00036 {
00037 public:
00038     /*!
00039      * Default constructor
00040      */
00041     Agnss_Ref_Time() = default;
00042
00043     double tow{};
00044     double week{};
00045     double seconds{};
00046     double microseconds{};
00047     bool valid{};
00048
00049     template <class Archive>
00050
00051     /*!
00052      * \brief Serialize is a boost standard method to be called by the boost XML
00053      * serialization. Here is used to save the ref time data on disk file.
00054      */
00055     inline void serialize(Archive& archive, const unsigned int version)
00056     {
00057         if (version)
00058             {
00059             };
00060         archive& BOOST_SERIALIZATION_NVP(tow);
00061         archive& BOOST_SERIALIZATION_NVP(week);
00062         archive& BOOST_SERIALIZATION_NVP(seconds);
00063         archive& BOOST_SERIALIZATION_NVP(microseconds);
00064         archive& BOOST_SERIALIZATION_NVP(valid);
00065     }
00066 };
00067
```

```
00068
00069 /*!
00070  * \brief Parses GNSS-SDR.AGNSS_ref_utc_time ("DD/MM/YYYY HH:MM:SS", UTC).
00071  * An empty string is not an error: the host's current time is returned, marked
00072  * valid. malformed_year and malformed_format (both optional) tell why a
00073  * non-empty string was rejected. Single parsing point for ControlThread and
00074  * SatelliteVisibility.
00075  */
00076 inline Agnss_Ref_Time parse_agnss_ref_utc_time(const std::string& ref_time_str, bool* malformed_year =
      nullptr, bool* malformed_format = nullptr)
00077 {
00078     if (malformed_year != nullptr)
00079         {
00080             *malformed_year = false;
00081         }
00082     if (malformed_format != nullptr)
00083         {
00084             *malformed_format = false;
00085         }
00086     Agnss_Ref_Time result{};
00087     if (ref_time_str.empty())
00088         {
00089             result.seconds = static_cast<double>(time(nullptr));
00090             result.valid = true;
00091             return result;
00092         }
00093     struct tm tm{};
00094     if (strptime(ref_time_str.c_str(), "%d/%m/%Y %H:%M:%S", &tm) != nullptr)
00095         {
00096             const time_t parsed = timegm(&tm);
00097             if (parsed > 0)
00098                 {
00099                     result.seconds = static_cast<double>(parsed);
00100                     result.valid = true;
00101                 }
00102             else if (malformed_year != nullptr)
00103                 {
00104                     *malformed_year = true;
00105                 }
00106         }
00107     else if (malformed_format != nullptr)
00108         {
00109             *malformed_format = true;
00110         }
00111     return result;
00112 }
00113
00114
00115 /** \} */
00116 /** \} */
00117 #endif  // GNSS_SDR_AGNSS_REF_TIME_H
```

13.636 Beidou_B1C.h File Reference

Defines system parameters for BeiDou B1C open service signal.

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double BEIDOU_B1C_FREQ_HZ = FREQ1

  §4.2.2, Table 4-1 [Hz]
- constexpr double BEIDOU_B1C_CODE_RATE_CPS = 1.023e6

  §5.2.1 [chips/s]
- constexpr double BEIDOU_B1C_CODE_LENGTH_CHIPS = 10230.0

  §5.2.1 [chips]
- constexpr double BEIDOU_B1C_CODE_PERIOD_S = 0.01

  §5.2.1, 10 ms [s]
- constexpr uint32_t BEIDOU_B1C_CODE_PERIOD_MS = 10

  §5.2.1 [ms]
- constexpr uint32_t BEIDOU_B1C_OPT_ACQ_FS_SPS = 4000000

  4 samples/chip at 1.023 Mcps

- constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH = 1800

  §5.2.2 [chips]
- constexpr int32_t BEIDOU_B1C_TELEMETRY_RATE_SPS = 100

  §6.2 [symbols/s]
- constexpr int32_t BEIDOU_B1C_SYMBOLS_PER_BIT = 1

  100 sps, 10 ms/code period
- constexpr int32_t BEIDOU_B1C_FRAME_LENGTH_SYMBOLS = 1800

  §6.2 [symbols]
- constexpr int32_t BEIDOU_B1C_FRAME_PERIOD_S = 18

  §6.2 [s]
- constexpr int32_t BEIDOU_B1C_NUMBER_OF_PRNS = 63

  §5.2.1

13.636.1 Detailed Description

Defines system parameters for BeiDou B1C open service signal.

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

Parameter values from BDS-SIS-ICD-B1C-1.0 (2017-12).

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.

Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file Beidou_B1C.h.

13.637 Beidou_B1C.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B1C.h
00003  * \brief Defines system parameters for BeiDou B1C open service signal
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * Parameter values from BDS-SIS-ICD-B1C-1.0 (2017-12).
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018 #ifndef GNSS_SDR_BEIDOU_B1C_H
00019 #define GNSS_SDR_BEIDOU_B1C_H
00020
00021 #include "gnss_frequencies.h"
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029 constexpr double BEIDOU_B1C_FREQ_HZ = FREQ1;                      //!< §4.2.2, Table 4-1 [Hz]
00030 constexpr double BEIDOU_B1C_CODE_RATE_CPS = 1.023e6;              //!< §5.2.1 [chips/s]
00031 constexpr double BEIDOU_B1C_CODE_LENGTH_CHIPS = 10230.0;          //!< §5.2.1 [chips]
00032 constexpr double BEIDOU_B1C_CODE_PERIOD_S = 0.01;                 //!< §5.2.1, 10 ms [s]
00033 constexpr uint32_t BEIDOU_B1C_CODE_PERIOD_MS = 10;                //!< §5.2.1 [ms]
00034 constexpr uint32_t BEIDOU_B1C_OPT_ACQ_FS_SPS = 4000000;           //!< 4 samples/chip at 1.023 Mcps
00035 constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH = 1800;  //!< §5.2.2 [chips]
00036 constexpr int32_t BEIDOU_B1C_TELEMETRY_RATE_SPS = 100;            //!< §6.2 [symbols/s]
00037 constexpr int32_t BEIDOU_B1C_SYMBOLS_PER_BIT = 1;                //!< 100 sps, 10 ms/code period
00038 constexpr int32_t BEIDOU_B1C_FRAME_LENGTH_SYMBOLS = 1800;         //!< §6.2 [symbols]
00039 constexpr int32_t BEIDOU_B1C_FRAME_PERIOD_S = 18;                 //!< §6.2 [s]
00040 constexpr int32_t BEIDOU_B1C_NUMBER_OF_PRNS = 63;                 //!< §5.2.1
00041
00042 /** \} */
00043 /** \} */
00044 #endif  // GNSS_SDR_BEIDOU_B1C_H
```

13.638 Beidou_B1C_codes.h File Reference

Precomputed BeiDou B1C primary and secondary codes (declarations).
`#include <cstddef>`

Variables

- constexpr size_t BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr size_t BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH = 1800
- const char **BEIDOU_B1C_DATA_PRIMARY_CODE** [63][2559]
- const char **BEIDOU_B1C_PILOT_PRIMARY_CODE** [63][2559]
- const char **BEIDOU_B1C_PILOT_SECONDARY_CODE** [63][1801]

13.638.1 Detailed Description

Precomputed BeiDou B1C primary and secondary codes (declarations).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

Auto-generated by generate_beidou_b1c_codes.py. Do not edit manually. Definitions live in Beidou_B1C_codes.cc.
GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_B1C_codes.h.

13.638.2 Variable Documentation

13.638.2.1 BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH

`size_t BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH = 1800 [constexpr]`
Definition at line 25 of file Beidou_B1C_codes.h.

13.638.2.2 BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH

`size_t BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH = 2558 [constexpr]`
Definition at line 24 of file Beidou_B1C_codes.h.

13.639 Beidou_B1C_codes.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B1C_codes.h
00003  * \brief Precomputed BeiDou B1C primary and secondary codes (declarations)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * Auto-generated by generate_beidou_b1c_codes.py. Do not edit manually.
00007  * Definitions live in Beidou_B1C_codes.cc.
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019 #ifndef GNSS_SDR_BEIDOU_B1C_CODES_H
00020 #define GNSS_SDR_BEIDOU_B1C_CODES_H
00021
00022 #include <cstddef>
00023
00024 constexpr size_t BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH = 2558;
00025 constexpr size_t BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH = 1800;
00026
```

```
00027 extern const char BEIDOU_B1C_DATA_PRIMARY_CODE[63][2559];
00028
00029 extern const char BEIDOU_B1C_PILOT_PRIMARY_CODE[63][2559];
00030
00031 extern const char BEIDOU_B1C_PILOT_SECONDARY_CODE[63][1801];
00032
00033 #endif
```

13.640 Beidou_B1C_prn.h File Reference

B1C PRN Weil code parameters from BDS-SIS-ICD-B1C-1.0 Table 5-2/5-3/5-4.
`#include <cstdint>`

Variables

- constexpr int32_t BEIDOU_B1C_DATA_PRN_W [63]
- constexpr int32_t BEIDOU_B1C_DATA_PRN_P [63]
- constexpr int32_t BEIDOU_B1C_PILOT_PRN_W [63]
- constexpr int32_t BEIDOU_B1C_PILOT_PRN_P [63]
- constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_W [63]
- constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_P [63]

13.640.1 Detailed Description

B1C PRN Weil code parameters from BDS-SIS-ICD-B1C-1.0 Table 5-2/5-3/5-4.

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_B1C_prn.h.

13.640.2 Variable Documentation

13.640.2.1 BEIDOU_B1C_DATA_PRN_P

`int32_t BEIDOU_B1C_DATA_PRN_P[63] [constexpr]`
Definition at line 84 of file Beidou_B1C_prn.h.

13.640.2.2 BEIDOU_B1C_DATA_PRN_W

`int32_t BEIDOU_B1C_DATA_PRN_W[63] [constexpr]`
Definition at line 19 of file Beidou_B1C_prn.h.

13.640.2.3 BEIDOU_B1C_PILOT_PRN_P

`int32_t BEIDOU_B1C_PILOT_PRN_P[63] [constexpr]`
Definition at line 214 of file Beidou_B1C_prn.h.

13.640.2.4 BEIDOU_B1C_PILOT_PRN_W

`int32_t BEIDOU_B1C_PILOT_PRN_W[63] [constexpr]`
Definition at line 149 of file Beidou_B1C_prn.h.

13.640.2.5 BEIDOU_B1C_PILOT_SECONDARY_P

`int32_t BEIDOU_B1C_PILOT_SECONDARY_P[63] [constexpr]`
Definition at line 344 of file Beidou_B1C_prn.h.

13.640.2.6 BEIDOU_B1C_PILOT_SECONDARY_W

```
int32_t BEIDOU_B1C_PILOT_SECONDARY_W[63]  [constexpr]
```

Definition at line 279 of file Beidou_B1C_prn.h.

13.641 Beidou_B1C_prn.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B1C_prn.h
00003  * \brief B1C PRN Weil code parameters from BDS-SIS-ICD-B1C-1.0 Table 5-2/5-3/5-4
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_B1C_PRN_H
00017 #define GNSS_SDR_BEIDOU_B1C_PRN_H
00018 #include <cstdint>
00019 constexpr int32_t BEIDOU_B1C_DATA_PRN_W[63] = {
00020     2678,
00021     4802,
00022     958,
00023     859,
00024     3843,
00025     2232,
00026     124,
00027     4352,
00028     1816,
00029     1126,
00030     1860,
00031     4800,
00032     2267,
00033     424,
00034     4192,
00035     4333,
00036     2656,
00037     4148,
00038     243,
00039     1330,
00040     1593,
00041     1470,
00042     882,
00043     3202,
00044     5095,
00045     2546,
00046     1733,
00047     4795,
00048     4577,
00049     1627,
00050     3638,
00051     2553,
00052     3646,
00053     1087,
00054     1843,
00055     216,
00056     2245,
00057     726,
00058     1966,
00059     670,
00060     4130,
00061     53,
00062     4830,
00063     182,
00064     2181,
00065     2006,
00066     1080,
00067     2288,
00068     2027,
00069     271,
00070     915,
00071     497,
00072     139,
00073     3693,
00074     2054,
00075     4342,
```

```
00076     3342,
00077     2592,
00078     1007,
00079     310,
00080     4203,
00081     455,
00082     4318,
00083 };
00084 constexpr int32_t BEIDOU_B1C_DATA_PRN_P[63] = {
00085     699,
00086     694,
00087     7318,
00088     2127,
00089     715,
00090     6682,
00091     7850,
00092     5495,
00093     1162,
00094     7682,
00095     6792,
00096     9973,
00097     6596,
00098     2092,
00099     19,
00100     10151,
00101     6297,
00102     5766,
00103     2359,
00104     7136,
00105     1706,
00106     2128,
00107     6827,
00108     693,
00109     9729,
00110     1620,
00111     6805,
00112     534,
00113     712,
00114     1929,
00115     5355,
00116     6139,
00117     6339,
00118     1470,
00119     6867,
00120     7851,
00121     1162,
00122     7659,
00123     1156,
00124     2672,
00125     6043,
00126     2862,
00127     180,
00128     2663,
00129     6940,
00130     1645,
00131     1582,
00132     951,
00133     6878,
00134     7701,
00135     1823,
00136     2391,
00137     2606,
00138     822,
00139     6403,
00140     239,
00141     442,
00142     6769,
00143     2560,
00144     2502,
00145     5072,
00146     7268,
00147     341,
00148 };
00149 constexpr int32_t BEIDOU_B1C_PILOT_PRN_W[63] = {
00150     796,
00151     156,
00152     4198,
00153     3941,
00154     1374,
00155     1338,
00156     1833,
00157     2521,
00158     3175,
00159     168,
00160     2715,
00161     4408,
00162     3160,
```

```
00163     2796,
00164     459,
00165     3594,
00166     4813,
00167     586,
00168     1428,
00169     2371,
00170     2285,
00171     3377,
00172     4965,
00173     3779,
00174     4547,
00175     1646,
00176     1430,
00177     607,
00178     2118,
00179     4709,
00180     1149,
00181     3283,
00182     2473,
00183     1006,
00184     3670,
00185     1817,
00186     771,
00187     2173,
00188     740,
00189     1433,
00190     2458,
00191     3459,
00192     2155,
00193     1205,
00194     413,
00195     874,
00196     2463,
00197     1106,
00198     1590,
00199     3873,
00200     4026,
00201     4272,
00202     3556,
00203     128,
00204     1200,
00205     130,
00206     4494,
00207     1871,
00208     3073,
00209     4386,
00210     4098,
00211     1923,
00212     1176,
00213 };
00214 constexpr int32_t BEIDOU_B1C_PILOT_PRN_P[63] = {
00215     7575,
00216     2369,
00217     5688,
00218     539,
00219     2270,
00220     7306,
00221     6457,
00222     6254,
00223     5644,
00224     7119,
00225     1402,
00226     5557,
00227     5764,
00228     1073,
00229     7001,
00230     5910,
00231     10060,
00232     2710,
00233     1546,
00234     6887,
00235     1883,
00236     5613,
00237     5062,
00238     1038,
00239     10170,
00240     6484,
00241     1718,
00242     2535,
00243     1158,
00244     526,
00245     7331,
00246     5844,
00247     6423,
00248     6968,
00249     1280,
```

```
00250     1838,
00251     1989,
00252     6468,
00253     2091,
00254     1581,
00255     1453,
00256     6252,
00257     7122,
00258     7711,
00259     7216,
00260     2113,
00261     1095,
00262     1628,
00263     1713,
00264     6102,
00265     6123,
00266     6070,
00267     1115,
00268     8047,
00269     6795,
00270     2575,
00271     53,
00272     1729,
00273     6388,
00274     682,
00275     5565,
00276     7160,
00277     2277,
00278 };
00279 constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_W[63] = {
00280     269,
00281     1448,
00282     1028,
00283     1324,
00284     822,
00285     5,
00286     155,
00287     458,
00288     310,
00289     959,
00290     1238,
00291     1180,
00292     1288,
00293     334,
00294     885,
00295     1362,
00296     181,
00297     1648,
00298     838,
00299     313,
00300     750,
00301     225,
00302     1477,
00303     309,
00304     108,
00305     1457,
00306     149,
00307     322,
00308     271,
00309     576,
00310     1103,
00311     450,
00312     399,
00313     241,
00314     1045,
00315     164,
00316     513,
00317     687,
00318     422,
00319     303,
00320     324,
00321     495,
00322     725,
00323     780,
00324     367,
00325     882,
00326     631,
00327     37,
00328     647,
00329     1043,
00330     24,
00331     120,
00332     134,
00333     136,
00334     158,
00335     214,
00336     335,
```

```
00337     340,
00338     661,
00339     889,
00340     929,
00341     1002,
00342     1149,
00343 };
00344 constexpr int32_t BEIDOU_B1C_PILOT_SECONDARY_P[63] = {
00345     1889,
00346     1268,
00347     1593,
00348     1186,
00349     1239,
00350     1930,
00351     176,
00352     1696,
00353     26,
00354     1344,
00355     1271,
00356     1182,
00357     1381,
00358     1604,
00359     1333,
00360     1185,
00361     31,
00362     704,
00363     1190,
00364     1646,
00365     1385,
00366     113,
00367     860,
00368     1656,
00369     1921,
00370     1173,
00371     1928,
00372     57,
00373     150,
00374     1214,
00375     1148,
00376     1458,
00377     1519,
00378     1635,
00379     1257,
00380     1687,
00381     1382,
00382     1514,
00383     1,
00384     1583,
00385     1806,
00386     1664,
00387     1338,
00388     1111,
00389     1706,
00390     1543,
00391     1813,
00392     228,
00393     2871,
00394     2884,
00395     1823,
00396     75,
00397     11,
00398     63,
00399     1937,
00400     22,
00401     1768,
00402     1526,
00403     1402,
00404     1445,
00405     1680,
00406     1290,
00407     1245,
00408 };
00409 #endif
```

13.642 Beidou_B1I.h File Reference

Defines system parameters for BeiDou B1I signal and DNAV data.

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double BEIDOU_B1I_FREQ_HZ = FREQ1_BDS

  B1I [Hz].
- constexpr double BEIDOU_B1I_CODE_RATE_CPS = 2.046e6

  Beidou B1I code rate [chips/s].
- constexpr double BEIDOU_B1I_CODE_LENGTH_CHIPS = 2046.0

  Beidou B1I code length [chips].
- constexpr double BEIDOU_B1I_CODE_PERIOD_S = 0.001

  Beidou B1I code period [seconds].
- constexpr double BEIDOU_B1I_PREAMBLE_DURATION_S = 0.220
- constexpr double BEIDOU_B1I_OPT_ACQ_FS_SPS = 4e6

  ~2 samples/chip (same ratio as GPS L1 at 2 MHz)
- constexpr uint32_t BEIDOU_B1I_CODE_PERIOD_MS = 1

  Beidou B1I code period [ms].
- constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_BITS = 11
- constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS = 220
- constexpr int32_t BEIDOU_B1I_SECONDARY_CODE_LENGTH = 20
- constexpr int32_t BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22
- constexpr int32_t BEIDOU_B1I_PREAMBLE_DURATION_MS = 220
- constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND = 50
- constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2
- constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_↩ cast<uint32_t>(BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT) * BEIDOU_B1I_CODE_PERIOD_MS)
- constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B1I_TELEMETRY↩ _RATE_BITS_SECOND * BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT
- constexpr char BEIDOU_B1I_SECONDARY_CODE_STR [21] = "00000100110101001110"
- constexpr char BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR [23] = "11111100000011000011100"
- constexpr char BEIDOU_B1I_D2_SECONDARY_CODE_STR [3] = "00"

13.642.1 Detailed Description

Defines system parameters for BeiDou B1I signal and DNAV data.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

Damian Miralles, 2018. dmiralles2009@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file Beidou_B1I.h.

13.643 Beidou_B1I.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B1I.h
00003  * \brief  Defines system parameters for BeiDou B1I signal and DNAV data
00004  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00005  * \author Damian Miralles, 2018. dmiralles2009@gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
```

```
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_BEIDOU_B1I_H
00020 #define GNSS_SDR_BEIDOU_B1I_H
00021
00022 #include "gnss_frequencies.h"
00023 #include <cstdint>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 // carrier and code frequencies
00032 constexpr double BEIDOU_B1I_FREQ_HZ = FREQ1_BDS;          //!< B1I [Hz]
00033 constexpr double BEIDOU_B1I_CODE_RATE_CPS = 2.046e6;      //!< Beidou B1I code rate [chips/s]
00034 constexpr double BEIDOU_B1I_CODE_LENGTH_CHIPS = 2046.0;  //!< Beidou B1I code length [chips]
00035 constexpr double BEIDOU_B1I_CODE_PERIOD_S = 0.001;       //!< Beidou B1I code period [seconds]
00036 constexpr double BEIDOU_B1I_PREAMBLE_DURATION_S = 0.220;
00037 constexpr double BEIDOU_B1I_OPT_ACQ_FS_SPS = 4e6;  //!< ~2 samples/chip (same ratio as GPS L1 at 2
     MHz)
00038 constexpr uint32_t BEIDOU_B1I_CODE_PERIOD_MS = 1;  //!< Beidou B1I code period [ms]
00039 constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_BITS = 11;
00040 constexpr uint32_t BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS = 220;
00041 constexpr int32_t BEIDOU_B1I_SECONDARY_CODE_LENGTH = 20;
00042 constexpr int32_t BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22;
00043 constexpr int32_t BEIDOU_B1I_PREAMBLE_DURATION_MS = 220;
00044 constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND = 50;
00045 constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT = 20;
00046 constexpr int32_t BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2;
00047 constexpr int32_t BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS =
     static_cast<int32_t>(static_cast<uint32_t>(BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT) *
     BEIDOU_B1I_CODE_PERIOD_MS);
00048 constexpr int32_t BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND *
     BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT;
00049 constexpr char BEIDOU_B1I_SECONDARY_CODE_STR[21] = "00000100110101001110";
00050 constexpr char BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR[23] = "11111100000011000001100";
00051 constexpr char BEIDOU_B1I_D2_SECONDARY_CODE_STR[3] = "00";
00052
00053
00054 /** \} */
00055 /** \} */
00056 #endif  // GNSS_SDR_BEIDOU_B1I_H
```

13.644 Beidou_B2a.h File Reference

System parameters for BeiDou B2a (BDS-SIS-ICD-B2a-1.0).

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double BEIDOU_B2A_FREQ_HZ = FREQ5

  BeiDou B2a [Hz].
- constexpr double BEIDOU_B2A_CODE_RATE_CPS = 10.23e6

  B2a primary code rate [chips/s].
- constexpr double BEIDOU_B2A_CODE_LENGTH_CHIPS = 10230.0

  B2a primary code length [chips].
- constexpr double BEIDOU_B2A_CODE_PERIOD_S = 0.001

  B2a primary code period [seconds].
- constexpr double BEIDOU_B2A_OPT_ACQ_FS_SPS = 20e6
- constexpr uint32_t BEIDOU_B2A_CODE_PERIOD_MS = 1
- constexpr uint32_t BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS = 24
- constexpr int32_t BEIDOU_B2A_SYMBOLS_PER_BIT = 1
- constexpr int32_t BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND = 1000
- constexpr int32_t BEIDOU_B2A_SECONDARY_CODE_LENGTH = 5
- constexpr char BEIDOU_B2A_SECONDARY_CODE_STR [6] = "00010"

ICD data secondary code, MSB first.

- constexpr char BEIDOU_B2A_PREAMBLE_SYMBOLS_STR [25] = "111000100100110111101000"

  0xE24DE8

13.644.1 Detailed Description

System parameters for BeiDou B2a (BDS-SIS-ICD-B2a-1.0).

Author

huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_B2a.h.

13.645 Beidou_B2a.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B2a.h
00003  * \brief System parameters for BeiDou B2a (BDS-SIS-ICD-B2a-1.0)
00004  * \author huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_B2A_H
00018 #define GNSS_SDR_BEIDOU_B2A_H
00019
00020 #include "gnss_frequencies.h"
00021 #include <cstdint>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028 constexpr double BEIDOU_B2A_FREQ_HZ = FREQ5;              //!< BeiDou B2a [Hz]
00029 constexpr double BEIDOU_B2A_CODE_RATE_CPS = 10.23e6;      //!< B2a primary code rate [chips/s]
00030 constexpr double BEIDOU_B2A_CODE_LENGTH_CHIPS = 10230.0;  //!< B2a primary code length [chips]
00031 constexpr double BEIDOU_B2A_CODE_PERIOD_S = 0.001;        //!< B2a primary code period [seconds]
00032 constexpr double BEIDOU_B2A_OPT_ACQ_FS_SPS = 20e6;
00033 constexpr uint32_t BEIDOU_B2A_CODE_PERIOD_MS = 1;
00034 constexpr uint32_t BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS = 24;
00035 constexpr int32_t BEIDOU_B2A_SYMBOLS_PER_BIT = 1;
00036 constexpr int32_t BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND = 1000;
00037 constexpr int32_t BEIDOU_B2A_SECONDARY_CODE_LENGTH = 5;
00038 constexpr char BEIDOU_B2A_SECONDARY_CODE_STR[6] = "00010";                        //!< ICD data
     secondary code, MSB first
00039 constexpr char BEIDOU_B2A_PREAMBLE_SYMBOLS_STR[25] = "111000100100110111101000";  //!< 0xE24DE8
00040
00041 /** \} */
00042 /** \} */
00043 #endif  // GNSS_SDR_BEIDOU_B2A_H
```

13.646 Beidou_B3I.h File Reference

Defines system parameters for BeiDou B3I signal and DNAV data.

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double BEIDOU_B3I_FREQ_HZ = FREQ3_BDS

  BeiDou B3I [Hz].
- constexpr double BEIDOU_B3I_CODE_RATE_CPS = 10.23e6

  BeiDou B3I code rate [chips/s].
- constexpr double BEIDOU_B3I_CODE_LENGTH_CHIPS = 10230.0

  BeiDou B3I code length [chips].
- constexpr double BEIDOU_B3I_CODE_PERIOD_S = 0.001

  BeiDou B3I code period [seconds].
- constexpr double BEIDOU_B3I_PREAMBLE_DURATION_S = 0.220
- constexpr double BEIDOU_B3I_OPT_ACQ_FS_SPS = 10e6
- constexpr uint32_t BEIDOU_B3I_CODE_PERIOD_MS = 1

  BeiDou B3I code period [ms].
- constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_BITS = 11
- constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS = 220
- constexpr int32_t BEIDOU_B3I_SECONDARY_CODE_LENGTH = 20
- constexpr int32_t BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22
- constexpr int32_t BEIDOU_B3I_PREAMBLE_DURATION_MS = 220
- constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND = 50

  D1 NAV message bit rate [bits/s].
- constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2
- constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS = static_cast<int32_t>(static_↩ cast<uint32_t>(BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT) * BEIDOU_B3I_CODE_PERIOD_MS)
- constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B3I_TELEMETRY_RATE_BITS_SECO * BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT
- constexpr char BEIDOU_B3I_SECONDARY_CODE_STR [21] = "00000100110101001110"
- constexpr char BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR [23] = "11111100000011000011100"
- constexpr char BEIDOU_B3I_D2_SECONDARY_CODE_STR [3] = "00"

13.646.1 Detailed Description

Defines system parameters for BeiDou B3I signal and DNAV data.

Author

Damian Miralles, 2019. dmiralles2009@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_B3I.h.

13.647 Beidou_B3I.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_B3I.h
00003  * \brief  Defines system parameters for BeiDou B3I signal and DNAV data
00004  * \author Damian Miralles, 2019. dmiralles2009@gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
```

```
00017
00018 #ifndef GNSS_SDR_BEIDOU_B3I_H
00019 #define GNSS_SDR_BEIDOU_B3I_H
00020
00021 #include "gnss_frequencies.h"
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 // carrier and code frequencies
00031 constexpr double BEIDOU_B3I_FREQ_HZ = FREQ3_BDS;          //!< BeiDou B3I [Hz]
00032 constexpr double BEIDOU_B3I_CODE_RATE_CPS = 10.23e6;      //!< BeiDou B3I code rate [chips/s]
00033 constexpr double BEIDOU_B3I_CODE_LENGTH_CHIPS = 10230.0;  //!< BeiDou B3I code length [chips]
00034 constexpr double BEIDOU_B3I_CODE_PERIOD_S = 0.001;        //!< BeiDou B3I code period [seconds]
00035 constexpr double BEIDOU_B3I_PREAMBLE_DURATION_S = 0.220;
00036 constexpr double BEIDOU_B3I_OPT_ACQ_FS_SPS = 10e6;
00037 constexpr uint32_t BEIDOU_B3I_CODE_PERIOD_MS = 1;  //!< BeiDou B3I code period [ms]
00038 constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_BITS = 11;
00039 constexpr uint32_t BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS = 220;  // **************
00040 constexpr int32_t BEIDOU_B3I_SECONDARY_CODE_LENGTH = 20;
00041 constexpr int32_t BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS = 22;
00042 constexpr int32_t BEIDOU_B3I_PREAMBLE_DURATION_MS = 220;
00043 constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND = 50;  //!< D1 NAV message bit rate [bits/s]
00044 constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT = 20;
00045 constexpr int32_t BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT = 2;  // *************
00046 constexpr int32_t BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS =
      static_cast<int32_t>(static_cast<uint32_t>(BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT) *
      BEIDOU_B3I_CODE_PERIOD_MS);
00047 constexpr int32_t BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND = BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND *
      BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT;
00048 constexpr char BEIDOU_B3I_SECONDARY_CODE_STR[21] = "00000100110101001110";
00049 constexpr char BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR[23] = "11111100000011000001100";
00050 constexpr char BEIDOU_B3I_D2_SECONDARY_CODE_STR[3] = "00";
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_BEIDOU_B3I_H
```

13.648 beidou_bdgim.h File Reference

BeiDou Global Ionospheric delay Model (BDGIM), BDS-SIS-ICD-B1C-1.0 §7.8.

`#include <cstdint>`

Functions

- double beidou_bdgim_delay_m (double time_mjd, double lat_rad, double lon_rad, double az_rad, double el_rad, const double alpha[9], double freq_hz)

  Compute BDGIM slant ionospheric delay [m] at carrier frequency freq_hz.
- double beidou_bdgim_nearest_odd_hour_mjd (double time_mjd)

  ICD §7.8.2: odd hour of day (01:00 . . . 23:00) nearest to time_mjd [MJD days].

13.648.1 Detailed Description

BeiDou Global Ionospheric delay Model (BDGIM), BDS-SIS-ICD-B1C-1.0 §7.8.

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_bdgim.h.

13.649 beidou_bdgim.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_bdgim.h
00003  * \brief BeiDou Global Ionospheric delay Model (BDGIM), BDS-SIS-ICD-B1C-1.0 §7.8
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_BDGIM_H
00017 #define GNSS_SDR_BEIDOU_BDGIM_H
00018
00019 #include <cstdint>
00020
00021 /** \addtogroup Core
00022  * \{ */
00023 /** \addtogroup System_Parameters
00024  * \{ */
00025
00026 /*!
00027  * \brief Compute BDGIM slant ionospheric delay [m] at carrier frequency freq_hz.
00028  * \param time_mjd Modified Julian Date (days) of the computation epoch
00029  * \param lat_rad User geodetic latitude [rad]
00030  * \param lon_rad User geodetic longitude [rad]
00031  * \param az_rad Satellite azimuth [rad]
00032  * \param el_rad Satellite elevation [rad]
00033  * \param alpha Broadcast BDGIM parameters alpha1..alpha9 [TECu]
00034  * \param freq_hz Carrier frequency [Hz]
00035  * \return Slant ionospheric delay [m], or 0 if inputs are invalid
00036  */
00037 double beidou_bdgim_delay_m(
00038     double time_mjd,
00039     double lat_rad,
00040     double lon_rad,
00041     double az_rad,
00042     double el_rad,
00043     const double alpha[9],
00044     double freq_hz);
00045
00046 /*!
00047  * \brief ICD §7.8.2: odd hour of day (01:00 ... 23:00) nearest to time_mjd [MJD days].
00048  */
00049 double beidou_bdgim_nearest_odd_hour_mjd(double time_mjd);
00050
00051 /** \} */
00052 /** \} */
00053 #endif  // GNSS_SDR_BEIDOU_BDGIM_H
```

13.650 Beidou_CNAV1.h File Reference

B-CNAV1 navigation message constants (BDS-SIS-ICD-B1C-1.0 §6.2, §7).

```
#include "MATH_CONSTANTS.h"
#include <cstdint>
```

Variables

- constexpr int32_t BEIDOU_CNAV1_FRAME_SYMBOLS = 1800
- constexpr int32_t BEIDOU_CNAV1_FRAME_PERIOD_S = 18
- constexpr int32_t BEIDOU_CNAV1_SYMBOL_RATE_SPS = 100
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME1_SYMBOLS = 72
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME2_SYMBOLS = 1200
- constexpr int32_t BEIDOU_CNAV1_SUBFRAME3_SYMBOLS = 528
- constexpr int32_t BEIDOU_CNAV1_SOH_LSB_S = 18
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVED_SYMBOLS = 1728
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_ROWS = 36
- constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_COLS = 48
- constexpr int32_t BEIDOU_CNAV1_SF2_DATA_BITS = 600

- constexpr int32_t BEIDOU_CNAV1_SF3_DATA_BITS = 264
- constexpr int32_t BEIDOU_CNAV1_CRC_BITS = 24
- constexpr int32_t BEIDOU_CNAV1_PAGE_IONO_UTC = 1
- constexpr int32_t BDS_EPH_SOURCE_CNAV1 = 7

  RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV1 (DNAV uses 0–6).
- constexpr double BEIDOU_CNAV1_A_REF_MEO = 27906100.0
- constexpr double BEIDOU_CNAV1_A_REF_IGSO = 42162200.0
- constexpr double BEIDOU_CNAV1_TOE_TOC_LSB = 300.0
- constexpr double BEIDOU_CNAV1_DELTA_A_LSB = TWO_N9
- constexpr double BEIDOU_CNAV1_A_DOT_LSB = TWO_N21

  2^-21 m/s, Table 7-8
- constexpr double BEIDOU_CNAV1_DELTA_N0_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI
- constexpr double BEIDOU_CNAV1_M0_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_E_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_OMEGA_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_I0_LSB = TWO_N32 * GNSS_PI
- constexpr double BEIDOU_CNAV1_OMEGADOT_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_IDOT_LSB = TWO_N44 * GNSS_PI
- constexpr double BEIDOU_CNAV1_CIS_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CIC_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CRS_LSB = TWO_N8
- constexpr double BEIDOU_CNAV1_CRC_LSB = TWO_N8
- constexpr double BEIDOU_CNAV1_CUS_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_CUC_LSB = TWO_N30
- constexpr double BEIDOU_CNAV1_AF0_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_AF1_LSB = TWO_N50
- constexpr double BEIDOU_CNAV1_AF2_LSB = TWO_N66
- constexpr double BEIDOU_CNAV1_TGD_LSB = TWO_N34
- constexpr double BEIDOU_CNAV1_ISC_LSB = TWO_N34

13.650.1 Detailed Description

B-CNAV1 navigation message constants (BDS-SIS-ICD-B1C-1.0 §6.2, §7).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_CNAV1.h.

13.651 Beidou_CNAV1.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_CNAV1.h
00003  * \brief B-CNAV1 navigation message constants (BDS-SIS-ICD-B1C-1.0 §6.2, §7)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
```

```
00017 #ifndef GNSS_SDR_BEIDOU_CNAV1_H
00018 #define GNSS_SDR_BEIDOU_CNAV1_H
00019
00020 #include "MATH_CONSTANTS.h"
00021 #include <cstdint>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028 constexpr int32_t BEIDOU_CNAV1_FRAME_SYMBOLS = 1800;
00029 constexpr int32_t BEIDOU_CNAV1_FRAME_PERIOD_S = 18;
00030 constexpr int32_t BEIDOU_CNAV1_SYMBOL_RATE_SPS = 100;
00031 constexpr int32_t BEIDOU_CNAV1_SUBFRAME1_SYMBOLS = 72;
00032 constexpr int32_t BEIDOU_CNAV1_SUBFRAME2_SYMBOLS = 1200;
00033 constexpr int32_t BEIDOU_CNAV1_SUBFRAME3_SYMBOLS = 528;
00034 constexpr int32_t BEIDOU_CNAV1_SOH_LSB_S = 18;
00035 constexpr int32_t BEIDOU_CNAV1_INTERLEAVED_SYMBOLS = 1728;
00036 constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_ROWS = 36;
00037 constexpr int32_t BEIDOU_CNAV1_INTERLEAVE_COLS = 48;
00038 constexpr int32_t BEIDOU_CNAV1_SF2_DATA_BITS = 600;
00039 constexpr int32_t BEIDOU_CNAV1_SF3_DATA_BITS = 264;
00040 constexpr int32_t BEIDOU_CNAV1_CRC_BITS = 24;
00041
00042 constexpr int32_t BEIDOU_CNAV1_PAGE_IONO_UTC = 1;
00043
00044 //! RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV1 (DNAV uses 0-6).
00045 constexpr int32_t BDS_EPH_SOURCE_CNAV1 = 7;
00046
00047 constexpr double BEIDOU_CNAV1_A_REF_MEO = 27906100.0;
00048 constexpr double BEIDOU_CNAV1_A_REF_IGSO = 42162200.0;
00049 constexpr double BEIDOU_CNAV1_TOE_TOC_LSB = 300.0;
00050 constexpr double BEIDOU_CNAV1_DELTA_A_LSB = TWO_N9;
00051 constexpr double BEIDOU_CNAV1_A_DOT_LSB = TWO_N21;  //!< 2^-21 m/s, Table 7-8
00052 constexpr double BEIDOU_CNAV1_DELTA_N0_LSB = TWO_N44 * GNSS_PI;
00053 constexpr double BEIDOU_CNAV1_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI;
00054 constexpr double BEIDOU_CNAV1_M0_LSB = TWO_N32 * GNSS_PI;
00055 constexpr double BEIDOU_CNAV1_E_LSB = TWO_N34;
00056 constexpr double BEIDOU_CNAV1_OMEGA_LSB = TWO_N32 * GNSS_PI;
00057 constexpr double BEIDOU_CNAV1_I0_LSB = TWO_N32 * GNSS_PI;
00058 constexpr double BEIDOU_CNAV1_OMEGADOT_LSB = TWO_N44 * GNSS_PI;
00059 constexpr double BEIDOU_CNAV1_IDOT_LSB = TWO_N44 * GNSS_PI;
00060 constexpr double BEIDOU_CNAV1_CIS_LSB = TWO_N30;
00061 constexpr double BEIDOU_CNAV1_CIC_LSB = TWO_N30;
00062 constexpr double BEIDOU_CNAV1_CRS_LSB = TWO_N8;
00063 constexpr double BEIDOU_CNAV1_CRC_LSB = TWO_N8;
00064 constexpr double BEIDOU_CNAV1_CUS_LSB = TWO_N30;
00065 constexpr double BEIDOU_CNAV1_CUC_LSB = TWO_N30;
00066 constexpr double BEIDOU_CNAV1_AF0_LSB = TWO_N34;
00067 constexpr double BEIDOU_CNAV1_AF1_LSB = TWO_N50;
00068 constexpr double BEIDOU_CNAV1_AF2_LSB = TWO_N66;
00069 constexpr double BEIDOU_CNAV1_TGD_LSB = TWO_N34;
00070 constexpr double BEIDOU_CNAV1_ISC_LSB = TWO_N34;
00071
00072 /** \} */
00073 /** \} */
00074 #endif  // GNSS_SDR_BEIDOU_CNAV1_H
```

13.652 beidou_cnav1_ephemeris.h File Reference

BeiDou B-CNAV1 ephemeris storage (BDS-SIS-ICD-B1C-1.0 §7).

```
#include "Beidou_CNAV1.h"
#include "gnss_ephemeris.h"
#include <boost/serialization/nvp.hpp>
```

Classes

- class Beidou_Cnav1_Ephemeris

13.652.1 Detailed Description

BeiDou B-CNAV1 ephemeris storage (BDS-SIS-ICD-B1C-1.0 §7).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_cnav1_ephemeris.h.

13.653 beidou_cnav1_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav1_ephemeris.h
00003  * \brief BeiDou B-CNAV1 ephemeris storage (BDS-SIS-ICD-B1C-1.0 §7)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_CNAV1_EPHEMERIS_H
00017 #define GNSS_SDR_BEIDOU_CNAV1_EPHEMERIS_H
00018
00019 #include "Beidou_CNAV1.h"
00020 #include "gnss_ephemeris.h"
00021 #include <boost/serialization/nvp.hpp>
00022
00023 class Beidou_Cnav1_Ephemeris : public Gnss_Ephemeris
00024 {
00025 public:
00026     Beidou_Cnav1_Ephemeris()
00027     {
00028         this->System = 'C';
00029     }
00030
00031     double TGD_B1Cp{};                      //!< §7.6 B1C pilot group delay
00032     double TGD_B2ap{};                      //!< §7.6 / B-CNAV2 MT30 B2a pilot group delay
00033     double ISC_B1Cd{};                      //!< §7.6 B1C data-to-pilot intra-frequency delay
      correction
00034     double ISC_B2ad{};                      //!< B-CNAV2 MT30 B2a data-to-pilot ISC
00035     double IODC{};                          //!< §7.4.2
00036     double IODE{};                          //!< §7.4.1
00037     int32_t hs{};                           //!< SF3 satellite health (2 bits, ICD §6.2.3)
00038     int32_t sig_type{BDS_EPH_SOURCE_CNAV1}; //!< RTKLIB eph.code for B-CNAV1
00039     int32_t nav_type{1};                    //!< 0: GEO, 1: MEO/IGSO
00040     int32_t sat_type{};                     //!< SF2 SatType (2 bits, ICD §7.5): 1=GEO, 2=IGSO,
      3=MEO. Same encoding as RINEX 4 CNV1
00041
00042     template <class Archive>
00043     void serialize(Archive& archive, const unsigned int version)
00044     {
00045         if (version)
00046             {
00047             };
00048         archive& BOOST_SERIALIZATION_NVP(PRN);
00049         archive& BOOST_SERIALIZATION_NVP(M_0);
00050         archive& BOOST_SERIALIZATION_NVP(delta_n);
00051         archive& BOOST_SERIALIZATION_NVP(delta_ndot);
00052         archive& BOOST_SERIALIZATION_NVP(ecc);
00053         archive& BOOST_SERIALIZATION_NVP(A0);
00054         archive& BOOST_SERIALIZATION_NVP(Adot);  // Gnss_Ephemeris::Adot
00055         archive& BOOST_SERIALIZATION_NVP(OMEGA_0);
00056         archive& BOOST_SERIALIZATION_NVP(i_0);
00057         archive& BOOST_SERIALIZATION_NVP(omega);
00058         archive& BOOST_SERIALIZATION_NVP(OMEGAdot);
00059         archive& BOOST_SERIALIZATION_NVP(idot);
00060         archive& BOOST_SERIALIZATION_NVP(Cuc);
00061         archive& BOOST_SERIALIZATION_NVP(Cus);
00062         archive& BOOST_SERIALIZATION_NVP(Crc);
00063         archive& BOOST_SERIALIZATION_NVP(Crs);
00064         archive& BOOST_SERIALIZATION_NVP(Cic);
00065         archive& BOOST_SERIALIZATION_NVP(Cis);
00066         archive& BOOST_SERIALIZATION_NVP(toe);
00067         archive& BOOST_SERIALIZATION_NVP(toc);
00068         archive& BOOST_SERIALIZATION_NVP(af0);
00069         archive& BOOST_SERIALIZATION_NVP(af1);
00070         archive& BOOST_SERIALIZATION_NVP(af2);
```

```
00071         archive& BOOST_SERIALIZATION_NVP(WN);
00072         archive& BOOST_SERIALIZATION_NVP(tow);
00073         archive& BOOST_SERIALIZATION_NVP(TGD_B1Cp);
00074         archive& BOOST_SERIALIZATION_NVP(TGD_B2ap);
00075         archive& BOOST_SERIALIZATION_NVP(ISC_B1Cd);
00076         archive& BOOST_SERIALIZATION_NVP(ISC_B2ad);
00077         archive& BOOST_SERIALIZATION_NVP(IODC);
00078         archive& BOOST_SERIALIZATION_NVP(IODE);
00079         archive& BOOST_SERIALIZATION_NVP(hs);
00080         archive& BOOST_SERIALIZATION_NVP(sig_type);
00081         archive& BOOST_SERIALIZATION_NVP(nav_type);
00082         archive& BOOST_SERIALIZATION_NVP(sat_type);
00083     }
00084 };
00085
00086 #endif  // GNSS_SDR_BEIDOU_CNAV1_EPHEMERIS_H
```

13.654 beidou_cnav1_iono.h File Reference

BeiDou B-CNAV1 ionospheric model (§7.8).

Classes

- class Beidou_Cnav1_Iono

13.654.1 Detailed Description

BeiDou B-CNAV1 ionospheric model (§7.8).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_cnav1_iono.h.

13.655 beidou_cnav1_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav1_iono.h
00003  * \brief BeiDou B-CNAV1 ionospheric model (§7.8)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_CNAV1_IONO_H
00017 #define GNSS_SDR_BEIDOU_CNAV1_IONO_H
00018
00019 /** \addtogroup Core
00020  * \{ */
00021 /** \addtogroup System_Parameters
00022  * \{ */
00023
00024 class Beidou_Cnav1_Iono
00025 {
00026 public:
00027     double alpha1{};
00028     double alpha2{};
00029     double alpha3{};
00030     double alpha4{};
00031     double alpha5{};
00032     double alpha6{};
00033     double alpha7{};
00034     double alpha8{};
00035     double alpha9{};
```

```
00036     bool valid{};
00037 };
00038
00039 /** \} */
00040 /** \} */
00041 #endif  // GNSS_SDR_BEIDOU_CNAV1_IONO_H
```

13.656 beidou_cnav1_ldpc.h File Reference

B-CNAV1 NB-LDPC decoder (BDS-SIS-ICD-B1C-1.0 §6.2.2, Appendix).

```
#include "beidou_ldpc.h"
```

Typedefs

- using BeidouCnav1LdpcGraph = BeidouLdpcGraph

Functions

- bool **beidou_cnav1_ldpc_decode_200_100** (const float *symbol_llr, int32_t num_bits, uint8_t *info_↩ bits600)
- bool **beidou_cnav1_ldpc_decode_88_44** (const float *symbol_llr, int32_t num_bits, uint8_t *info_bits264)
- bool **beidou_cnav1_ldpc_decode_200_100_codeword** (const float *symbol_llr, int32_t num_bits, uint8_t *codeword_bits1200)
- bool **beidou_cnav1_ldpc_decode_88_44_codeword** (const float *symbol_llr, int32_t num_bits, uint8_↩ t *codeword_bits528)
- bool **beidou_cnav1_ldpc_init_graph** (int32_t num_checks, int32_t num_variables, int32_t row_weight, const uint16_t *icd_index, const uint8_t *icd_element, int32_t num_entries, BeidouCnav1LdpcGraph &graph)
- const BeidouCnav1LdpcGraph & **beidou_cnav1_ldpc_graph_200_100** ()
- const BeidouCnav1LdpcGraph & **beidou_cnav1_ldpc_graph_88_44** ()

Variables

- constexpr int32_t BEIDOU_CNAV1_LDPC_N200 = 200
- constexpr int32_t BEIDOU_CNAV1_LDPC_K200 = 100
- constexpr int32_t BEIDOU_CNAV1_LDPC_M200 = 100
- constexpr int32_t BEIDOU_CNAV1_LDPC_DC200 = 4
- constexpr int32_t BEIDOU_CNAV1_LDPC_N88 = 88
- constexpr int32_t BEIDOU_CNAV1_LDPC_K88 = 44
- constexpr int32_t BEIDOU_CNAV1_LDPC_M88 = 44
- constexpr int32_t BEIDOU_CNAV1_LDPC_DC88 = 4
- constexpr int32_t BEIDOU_CNAV1_LDPC_NM = BEIDOU_LDPC_NM
- constexpr int32_t BEIDOU_CNAV1_LDPC_MAX_ITER = BEIDOU_LDPC_MAX_ITER
- constexpr uint16_t BEIDOU_CNAV1_H100_200_INDEX [400]
- constexpr uint8_t BEIDOU_CNAV1_H100_200_ELEMENT [400]
- constexpr uint16_t BEIDOU_CNAV1_H44_88_INDEX [176]
- constexpr uint8_t BEIDOU_CNAV1_H44_88_ELEMENT [176]

13.656.1 Detailed Description

B-CNAV1 NB-LDPC decoder (BDS-SIS-ICD-B1C-1.0 §6.2.2, Appendix).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_cnav1_ldpc.h.

13.657 beidou_cnav1_ldpc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav1_ldpc.h
00003  * \brief B-CNAV1 NB-LDPC decoder (BDS-SIS-ICD-B1C-1.0 §6.2.2, Appendix)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_CNAV1_LDPC_DECODER_H
00018 #define GNSS_SDR_BEIDOU_CNAV1_LDPC_DECODER_H
00019
00020 #include "beidou_ldpc.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027 constexpr int32_t BEIDOU_CNAV1_LDPC_N200 = 200;
00028 constexpr int32_t BEIDOU_CNAV1_LDPC_K200 = 100;
00029 constexpr int32_t BEIDOU_CNAV1_LDPC_M200 = 100;
00030 constexpr int32_t BEIDOU_CNAV1_LDPC_DC200 = 4;
00031 constexpr int32_t BEIDOU_CNAV1_LDPC_N88 = 88;
00032 constexpr int32_t BEIDOU_CNAV1_LDPC_K88 = 44;
00033 constexpr int32_t BEIDOU_CNAV1_LDPC_M88 = 44;
00034 constexpr int32_t BEIDOU_CNAV1_LDPC_DC88 = 4;
00035 constexpr int32_t BEIDOU_CNAV1_LDPC_NM = BEIDOU_LDPC_NM;
00036 constexpr int32_t BEIDOU_CNAV1_LDPC_MAX_ITER = BEIDOU_LDPC_MAX_ITER;
00037
00038 /*
00039  * GF(2^6) LUTs follow BDS-SIS-ICD-B1C-1.0 Appendix mapping tables.
00040  * Primitive polynomial: p(x)=x^6+x+1.
00041  */
00042 constexpr uint16_t BEIDOU_CNAV1_H100_200_INDEX[400] = {
00043     11, 62, 102, 150, 9, 60, 100, 148, 0, 51, 142, 197, 22, 80, 116, 154, 4, 90,
00044     131, 177, 47, 95, 138, 191, 51, 79, 146, 195, 44, 75, 142, 190, 13, 57, 135,
00045     198, 24, 65, 120, 173, 6, 88, 129, 179, 7, 89, 130, 176, 6, 58, 106, 158, 8,
00046     60, 108, 160, 44, 92, 139, 188, 4, 56, 104, 156, 10, 61, 101, 149, 39, 87, 123,
00047     168, 15, 67, 105, 167, 50, 78, 145, 194, 17, 98, 151, 187, 46, 94, 137, 190,
00048     14, 66, 104, 166, 7, 59, 107, 159, 21, 83, 119, 153, 31, 87, 114, 167, 2, 49,
00049     140, 199, 12, 64, 106, 164, 40, 53, 132, 159, 19, 96, 149, 185, 16, 68, 112,
00050     168, 14, 58, 132, 199, 34, 69, 125, 162, 23, 75, 119, 175, 42, 96, 144, 192,
00051     8, 63, 103, 151, 23, 81, 117, 155, 24, 93, 111, 182, 20, 72, 116, 172, 17, 69,
00052     113, 169, 34, 82, 130, 182, 1, 53, 101, 153, 46, 73, 140, 188, 13, 65, 107,
00053     165, 2, 54, 102, 154, 18, 70, 114, 170, 26, 67, 122, 175, 29, 77, 125, 177,
00054     36, 84, 120, 169, 25, 94, 108, 183, 39, 89, 137, 185, 21, 73, 117, 173, 28,
00055     76, 124, 176, 36, 90, 138, 186, 33, 68, 124, 161, 12, 56, 134, 197, 29, 85,
00056     112, 165, 45, 93, 136, 189, 27, 64, 123, 172, 28, 84, 115, 164, 25, 66, 121,
00057     174, 37, 85, 121, 170, 3, 50, 141, 196, 48, 76, 147, 192, 35, 70, 126, 163,
00058     32, 80, 128, 180, 0, 52, 100, 152, 43, 52, 135, 158, 35, 83, 131, 183, 10,
00059     62, 110, 162, 19, 71, 115, 171, 15, 59, 133, 196, 33, 81, 129, 181, 41, 54,
00060     133, 156, 20, 82, 118, 152, 38, 86, 122, 171, 30, 78, 126, 178, 9, 61, 109,
00061     161, 26, 95, 109, 180, 45, 72, 143, 191, 1, 48, 143, 198, 40, 98, 146, 194,
00062     18, 99, 148, 184, 5, 57, 105, 157, 41, 99, 147, 195, 31, 79, 127, 179, 3, 55,
00063     103, 155, 22, 74, 118, 174, 37, 91, 139, 187, 5, 91, 128, 178, 30, 86, 113,
00064     166, 43, 97, 145, 193, 16, 97, 150, 186, 11, 63, 111, 163, 32, 71, 127, 160,
00065     42, 55, 134, 157, 38, 88, 136, 184, 47, 74, 141, 189, 49, 77, 144, 193, 27,
00066     92, 110, 181};
00067
00068
00069 constexpr uint8_t BEIDOU_CNAV1_H100_200_ELEMENT[400] = {
00070     35, 13, 51, 60, 1, 44, 53, 24, 1, 45, 15, 6, 45, 15, 6, 1, 1, 44, 53, 24, 1,
00071     45, 15, 6, 35, 46, 56, 15, 6, 1, 45, 15, 15, 6, 1, 45, 44, 53, 24, 1, 24, 1,
00072     44, 30, 1, 45, 15, 6, 30, 24, 1, 44, 24, 1, 44, 30, 45, 15, 6, 1, 17, 38, 49,
00073     11, 24, 1, 44, 30, 24, 1, 44, 53, 24, 1, 44, 53, 30, 24, 1, 44, 33, 42, 14,
00074     24, 33, 42, 14, 24, 45, 15, 6, 1, 1, 45, 15, 6, 30, 24, 1, 44, 24, 1, 44, 53,
00075     1, 44, 30, 24, 57, 25, 9, 41, 1, 45, 15, 6, 1, 45, 15, 6, 42, 36, 12, 57, 6,
00076     1, 45, 15, 24, 1, 44, 53, 24, 1, 44, 30, 1, 45, 15, 6, 1, 45, 15, 6, 44, 53,
00077     24, 1, 30, 24, 1, 44, 1, 44, 30, 24, 53, 24, 1, 44, 1, 44, 53, 24, 27, 28,
00078     30, 31, 53, 24, 1, 44, 24, 1, 44, 30, 45, 15, 6, 1, 30, 24, 1, 44, 1, 45, 15,
00079     6, 26, 22, 14, 2, 35, 13, 18, 60, 45, 15, 6, 1, 30, 1, 44, 7, 6, 1, 45, 15,
00080     6, 1, 45, 15, 53, 24, 1, 44, 24, 1, 44, 53, 30, 24, 1, 44, 1, 44, 30, 24, 44,
00081     53, 24, 1, 53, 24, 1, 44, 44, 30, 24, 1, 30, 24, 1, 44, 1, 44, 30, 24, 1, 44,
00082     30, 24, 41, 16, 29, 51, 1, 44, 30, 24, 38, 23, 22, 7, 44, 53, 24, 1, 1, 45,
00083     15, 6, 30, 24, 1, 44, 53, 24, 1, 44, 6, 1, 45, 15, 24, 1, 44, 53, 35, 46, 56,
```

```
00084     15, 5, 33, 42, 14, 54, 7, 38, 23, 1, 45, 15, 6, 44, 30, 24, 1, 6, 1, 45, 15,
00085     53, 24, 1, 44, 44, 53, 24, 1, 1, 44, 53, 24, 1, 44, 30, 24, 44, 30, 24, 1, 1,
00086     44, 53, 24, 45, 15, 6, 1, 6, 1, 45, 15, 1, 44, 53, 24, 42, 47, 37, 32, 51, 60,
00087     35, 13, 29, 28, 30, 31, 6, 1, 45, 15, 24, 1, 44, 53, 44, 53, 24, 1, 44, 30,
00088     24, 1, 38, 49, 11, 17, 44, 30, 24, 1, 24, 1, 44, 30, 24, 1, 44, 30, 1, 44, 53,
00089     24, 53, 24, 1, 44};
00090
00091
00092 constexpr uint16_t BEIDOU_CNAV1_H44_88_INDEX[176] = {
00093     14, 35, 56, 70, 11, 29, 55, 73, 13, 39, 53, 69, 15, 34, 57, 71, 1, 27, 45, 54,
00094     23, 41, 63, 87, 2, 20, 46, 68, 6, 24, 50, 61, 2, 26, 61, 79, 9, 33, 59, 77, 4,
00095     30, 48, 74, 22, 42, 59, 76, 12, 38, 52, 68, 23, 43, 58, 77, 19, 21, 63, 64, 11,
00096     25, 65, 82, 17, 39, 44, 75, 9, 35, 49, 72, 19, 29, 66, 84, 13, 36, 56, 82, 17,
00097     43, 67, 81, 22, 40, 62, 86, 3, 21, 47, 69, 10, 24, 64, 83, 0, 37, 70, 86, 5, 31,
00098     49, 75, 4, 40, 53, 84, 5, 41, 52, 85, 18, 28, 67, 85, 0, 26, 44, 55, 10, 28, 54,
00099     72, 7, 30, 50, 81, 1, 36, 71, 87, 16, 38, 45, 74, 8, 34, 48, 73, 8, 32, 58, 76,
00100     12, 37, 57, 83, 6, 31, 51, 80, 15, 33, 47, 79, 16, 42, 66, 80, 7, 25, 51, 60, 3,
00101     27, 60, 78, 14, 32, 46, 78, 18, 20, 62, 65};
00102
00103
00104 constexpr uint8_t BEIDOU_CNAV1_H44_88_ELEMENT[176] = {
00105     30, 24, 1, 44, 24, 1, 44, 30, 40, 32, 61, 18, 53, 24, 1, 44, 51, 60, 35, 13, 18,
00106     15, 32, 61, 15, 6, 1, 45, 30, 24, 1, 44, 6, 1, 45, 15, 45, 15, 6, 1, 1, 45, 15,
00107     6, 1, 44, 53, 24, 24, 1, 44, 53, 44, 30, 24, 1, 34, 33, 45, 36, 55, 9, 34, 3,
00108     1, 44, 53, 24, 61, 47, 20, 8, 53, 24, 1, 44, 15, 6, 1, 45, 13, 18, 60, 35, 45,
00109     15, 6, 1, 24, 1, 44, 53, 37, 32, 52, 47, 44, 53, 24, 1, 39, 36, 34, 33, 44, 35,
00110     31, 50, 12, 25, 36, 14, 15, 35, 46, 56, 53, 24, 1, 44, 1, 44, 53, 24, 24, 1, 44,
00111     30, 44, 30, 24, 1, 15, 6, 1, 45, 30, 24, 1, 44, 2, 50, 22, 14, 33, 42, 14, 5, 34,
00112     3, 55, 9, 44, 35, 61, 50, 15, 6, 1, 45, 45, 15, 6, 1, 1, 44, 30, 24, 6, 1, 45,
00113     15, 1, 44, 53, 24};
00114
00115 bool beidou_cnav1_ldpc_decode_200_100(const float* symbol_llr, int32_t num_bits, uint8_t*
      info_bits600);
00116 bool beidou_cnav1_ldpc_decode_88_44(const float* symbol_llr, int32_t num_bits, uint8_t* info_bits264);
00117 bool beidou_cnav1_ldpc_decode_200_100_codeword(const float* symbol_llr, int32_t num_bits, uint8_t*
      codeword_bits1200);
00118 bool beidou_cnav1_ldpc_decode_88_44_codeword(const float* symbol_llr, int32_t num_bits, uint8_t*
      codeword_bits528);
00119
00120 using BeidouCnav1LdpcGraph = BeidouLdpcGraph;
00121
00122 bool beidou_cnav1_ldpc_init_graph(
00123     int32_t num_checks,
00124     int32_t num_variables,
00125     int32_t row_weight,
00126     const uint16_t* icd_index,
00127     const uint8_t* icd_element,
00128     int32_t num_entries,
00129     BeidouCnav1LdpcGraph& graph);
00130
00131
00132 const BeidouCnav1LdpcGraph& beidou_cnav1_ldpc_graph_200_100();
00133 const BeidouCnav1LdpcGraph& beidou_cnav1_ldpc_graph_88_44();
00134
00135 /** \} */
00136 /** \} */
00137 #endif  // GNSS_SDR_BEIDOU_CNAV1_LDPC_DECODER_H
```

13.658 beidou_cnav1_navigation_message.h File Reference

B-CNAV1 navigation message parser (§6.2).

```
#include "beidou_cnav1_ephemeris.h"
#include "beidou_cnav1_iono.h"
#include "beidou_cnav1_utc_model.h"
#include <array>
#include <cstdint>
#include <string>
```

Classes

- struct Bds3_B1c_PageCommon
- struct Bds3_B1c_AlmanacReduced
- struct Bds3_B1c_AlmanacMedium
- struct Bds3_B1c_Eop
- struct Bds3_B1c_Bgto

- struct Bds3_B1c_PageData
- struct Beidou_Cnav1_PageData_Message
- class Beidou_Cnav1_Navigation_Message

13.658.1 Detailed Description

B-CNAV1 navigation message parser (§6.2).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_cnav1_navigation_message.h.

13.659 beidou_cnav1_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav1_navigation_message.h
00003  * \brief B-CNAV1 navigation message parser (§6.2)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_CNAV1_NAVIGATION_MESSAGE_H
00018 #define GNSS_SDR_BEIDOU_CNAV1_NAVIGATION_MESSAGE_H
00019
00020 #include "beidou_cnav1_ephemeris.h"
00021 #include "beidou_cnav1_iono.h"
00022 #include "beidou_cnav1_utc_model.h"
00023 #include <array>
00024 #include <cstdint>
00025 #include <string>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032 struct Bds3_B1c_PageCommon
00033 {
00034     int32_t page_id{};
00035     int32_t sismai{};
00036     int32_t hs{};
00037     bool dif{};
00038     bool sif{};
00039     bool aif{};
00040 };
00041
00042
00043 struct Bds3_B1c_AlmanacReduced
00044 {
00045     int32_t prn{};
00046     int32_t sat_type{};
00047     double delta_a_m{};
00048     double omega0_rad{};
00049     double phi0_rad{};
00050     int32_t health{};
00051 };
00052
00053
00054 struct Bds3_B1c_AlmanacMedium
00055 {
00056     int32_t prn{};
00057     int32_t sat_type{};
00058     int32_t wna{};
00059     int32_t toa_s{};
00060     double eccentricity{};
```

```
00061     double delta_i_rad{};
00062     double sqrt_a_m_sqrt{};
00063     double omega0_rad{};
00064     double omega_dot_rad_s{};
00065     double omega_rad{};
00066     double m0_rad{};
00067     double af0_s{};
00068     double af1_s_s{};
00069     int32_t health{};
00070 };
00071
00072
00073 struct Bds3_B1c_Eop
00074 {
00075     int32_t t_eop_s{};
00076     double pm_x_arcsec{};
00077     double pm_x_dot_arcsec_day{};
00078     double pm_y_arcsec{};
00079     double pm_y_dot_arcsec_day{};
00080     double delta_ut1_s{};
00081     double delta_ut1_dot_s_day{};
00082 };
00083
00084
00085 struct Bds3_B1c_Bgto
00086 {
00087     int32_t gnss_id{};
00088     int32_t wn0_bgto{};
00089     int32_t t0_bgto_s{};
00090     double a0_bgto_s{};
00091     double a1_bgto_s_s{};
00092     double a2_bgto_s_s2{};
00093 };
00094
00095
00096 struct Bds3_B1c_PageData
00097 {
00098     Bds3_B1c_PageCommon common{};
00099     int32_t sisaioc{};
00100     int32_t sisaioe{};
00101     int32_t wna{};
00102     int32_t toa_s{};
00103     std::array<Bds3_B1c_AlmanacReduced, 4> reduced_almanac{};
00104     Bds3_B1c_Eop eop{};
00105     Bds3_B1c_Bgto bgto{};
00106     Bds3_B1c_AlmanacMedium medium_almanac{};
00107 };
00108
00109
00110 struct Beidou_Cnav1_PageData_Message
00111 {
00112     int32_t PRN{};
00113     Bds3_B1c_PageData page_data{};
00114 };
00115
00116
00117 class Beidou_Cnav1_Navigation_Message
00118 {
00119 public:
00120     bool decode_frame(const float* symbols, int32_t num_symbols, int32_t expected_prn = 0, int32_t*
      fail_stage = nullptr);
00121     bool decode_frame_symbols(const float* symbols, int32_t num_symbols, int32_t expected_prn = 0,
      int32_t* fail_stage = nullptr);
00122     bool probe_subframe1_prn(const float* symbols, int32_t num_symbols, int32_t expected_prn) const;
00123     bool have_new_ephemeris() const;
00124     bool have_new_iono() const;
00125     bool have_new_utc_model() const;
00126     bool have_new_page_data() const;
00127     void clear_flags();
00128     const Beidou_Cnav1_Ephemeris& get_ephemeris() const;
00129     const Beidou_Cnav1_Iono& get_iono() const;
00130     const Beidou_Cnav1_Utc_Model& get_utc_model() const;
00131     const Bds3_B1c_PageData& get_page_data() const;
00132     //! Last successfully decoded frame as '0'/'1' chars for Nav_msg_from_TLM (SF1 info + SF2 [+
      SF3]).
00133     const std::string& get_last_nav_bits() const;
00134     double get_tow_s() const;
00135
00136 private:
00137     Beidou_Cnav1_Ephemeris ephemeris_{};
00138     Beidou_Cnav1_Iono iono_{};
00139     Beidou_Cnav1_Utc_Model utc_model_{};
00140     bool flag_new_ephemeris_{false};
00141     bool flag_new_iono_{false};
00142     bool flag_new_utc_{false};
00143     bool flag_new_page_data_{false};
00144     double tow_s_{0.0};
```

```
00145     Bds3_B1c_PageData page_data_{};
00146     std::string last_nav_bits_{};
00147 };
00148
00149 /** \} */
00150 /** \} */
00151 #endif  // GNSS_SDR_BEIDOU_CNAV1_NAVIGATION_MESSAGE_H
```

13.660 beidou_cnav1_utc_model.h File Reference

BeiDou B-CNAV1 UTC model (BDS-SIS-ICD-B1C-1.0 §7.12).

```
#include <cstdint>
```

Classes

- class Beidou_Cnav1_Utc_Model

13.660.1 Detailed Description

BeiDou B-CNAV1 UTC model (BDS-SIS-ICD-B1C-1.0 §7.12).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_cnav1_utc_model.h.

13.661 beidou_cnav1_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav1_utc_model.h
00003  * \brief BeiDou B-CNAV1 UTC model (BDS-SIS-ICD-B1C-1.0 §7.12)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016 #ifndef GNSS_SDR_BEIDOU_CNAV1_UTC_MODEL_H
00017 #define GNSS_SDR_BEIDOU_CNAV1_UTC_MODEL_H
00018
00019 #include <cstdint>
00020
00021 /** \addtogroup Core
00022  * \{ */
00023 /** \addtogroup System_Parameters
00024  * \{ */
00025
00026 class Beidou_Cnav1_Utc_Model
00027 {
00028 public:
00029     double A0{};
00030     double A1{};
00031     double A2{};
00032     int32_t tot{};
00033     int32_t WN_t{};
00034     int32_t WN_LSF{};
00035     int32_t DN{};
00036     int32_t delta_t_LS{};   //!< Table 7-20: leap seconds before new leap effective [s]
00037     int32_t delta_t_LSF{};  //!< Table 7-20: leap seconds after new leap effective [s]
00038     bool valid{};
00039 };
00040
00041 /** \} */
```

```
00042 /** \} */
00043 #endif  // GNSS_SDR_BEIDOU_CNAV1_UTC_MODEL_H
```

13.662 Beidou_CNAV2.h File Reference

B-CNAV2 navigation message constants (BDS-SIS-ICD-B2a-1.0).

```
#include "Beidou_CNAV1.h"
#include <cstdint>
```

Variables

- constexpr int32_t BEIDOU_CNAV2_FRAME_SYMBOLS = 600
- constexpr int32_t BEIDOU_CNAV2_FRAME_PERIOD_S = 3
- constexpr int32_t BEIDOU_CNAV2_SYMBOL_RATE_SPS = 200
- constexpr int32_t BEIDOU_CNAV2_MS_PER_SYMBOL = 5
- constexpr int32_t BEIDOU_CNAV2_FRAME_MS = 3000
- constexpr int32_t BEIDOU_CNAV2_PREAMBLE_SYMBOLS = 24
- constexpr int32_t BEIDOU_CNAV2_PREAMBLE_MS = 120

  24 symbols ∗ 5 ms secondary code
- constexpr int32_t BEIDOU_CNAV2_LDPC_SYMBOLS = 576

  encoded payload: 96 GF(64) symbols, 6 bits each
- constexpr int32_t BEIDOU_CNAV2_INFO_BITS = 288
- constexpr int32_t BEIDOU_CNAV2_CRC_BITS = 24
- constexpr int32_t BEIDOU_CNAV2_DATA_BITS = BEIDOU_CNAV2_INFO_BITS - BEIDOU_CNAV2_CRC↩ _BITS
- constexpr int32_t BEIDOU_CNAV2_MES_TYPE_BITS = 6
- constexpr int32_t BEIDOU_CNAV2_SOW_LSB_S = 3
- constexpr double BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD = 5.2

  on the normalized soft preamble statistic, about $|N(0,1)|$ on noise
- constexpr int32_t BEIDOU_CNAV2_MSG_EPH1 = 10
- constexpr int32_t BEIDOU_CNAV2_MSG_EPH2 = 11
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_IONO = 30
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_ALM = 31
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_EOP = 32
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_UTC = 33
- constexpr int32_t BEIDOU_CNAV2_MSG_CLK_DC = 34
- constexpr int32_t BEIDOU_CNAV2_MSG_ALM = 40
- constexpr int32_t BEIDOU_CNAV2_MT10_IODE_BIT = 53
- constexpr int32_t BEIDOU_CNAV2_IODE_BITS = 8
- constexpr int32_t BDS_EPH_SOURCE_CNAV2 = 9

  RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV2.

13.662.1 Detailed Description

B-CNAV2 navigation message constants (BDS-SIS-ICD-B2a-1.0).

Author

huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Beidou_CNAV2.h.

13.663 Beidou_CNAV2.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_CNAV2.h
00003  * \brief B-CNAV2 navigation message constants (BDS-SIS-ICD-B2a-1.0).
00004  * \author huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_CNAV2_H
00018 #define GNSS_SDR_BEIDOU_CNAV2_H
00019
00020 #include "Beidou_CNAV1.h"
00021 #include <cstdint>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028 constexpr int32_t BEIDOU_CNAV2_FRAME_SYMBOLS = 600;
00029 constexpr int32_t BEIDOU_CNAV2_FRAME_PERIOD_S = 3;
00030 constexpr int32_t BEIDOU_CNAV2_SYMBOL_RATE_SPS = 200;
00031 constexpr int32_t BEIDOU_CNAV2_MS_PER_SYMBOL = 5;
00032 constexpr int32_t BEIDOU_CNAV2_FRAME_MS = 3000;
00033 constexpr int32_t BEIDOU_CNAV2_PREAMBLE_SYMBOLS = 24;
00034 constexpr int32_t BEIDOU_CNAV2_PREAMBLE_MS = 120;   //!< 24 symbols * 5 ms secondary code
00035 constexpr int32_t BEIDOU_CNAV2_LDPC_SYMBOLS = 576;  //!< encoded payload: 96 GF(64) symbols, 6 bits
     each
00036 constexpr int32_t BEIDOU_CNAV2_INFO_BITS = 288;
00037 constexpr int32_t BEIDOU_CNAV2_CRC_BITS = 24;
00038 constexpr int32_t BEIDOU_CNAV2_DATA_BITS = BEIDOU_CNAV2_INFO_BITS - BEIDOU_CNAV2_CRC_BITS;
00039 constexpr int32_t BEIDOU_CNAV2_MES_TYPE_BITS = 6;
00040 constexpr int32_t BEIDOU_CNAV2_SOW_LSB_S = 3;
00041 constexpr double BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD = 5.2;  //!< on the normalized soft
     preamble statistic, about |N(0,1)| on noise
00042
00043 constexpr int32_t BEIDOU_CNAV2_MSG_EPH1 = 10;
00044 constexpr int32_t BEIDOU_CNAV2_MSG_EPH2 = 11;
00045 constexpr int32_t BEIDOU_CNAV2_MSG_CLK_IONO = 30;
00046 constexpr int32_t BEIDOU_CNAV2_MSG_CLK_ALM = 31;
00047 constexpr int32_t BEIDOU_CNAV2_MSG_CLK_EOP = 32;
00048 constexpr int32_t BEIDOU_CNAV2_MSG_CLK_UTC = 33;
00049 constexpr int32_t BEIDOU_CNAV2_MSG_CLK_DC = 34;
00050 constexpr int32_t BEIDOU_CNAV2_MSG_ALM = 40;
00051
00052 // Broadcast IODE: 8 bits immediately before toe in MT10. MT11 has no IODE.
00053 constexpr int32_t BEIDOU_CNAV2_MT10_IODE_BIT = 53;
00054 constexpr int32_t BEIDOU_CNAV2_IODE_BITS = 8;
00055
00056 //! RTKLIB eph_t::code / Beidou_Cnav1_Ephemeris::sig_type for B-CNAV2.
00057 constexpr int32_t BDS_EPH_SOURCE_CNAV2 = 9;
00058
00059 /** \} */
00060 /** \} */
00061 #endif  // GNSS_SDR_BEIDOU_CNAV2_H
```

13.664 beidou_cnav2_ldpc.h File Reference

B-CNAV2 LDPC(96,48) decoder (BDS-SIS-ICD-B2a-1.0 section 6.2.2).

```
#include "beidou_ldpc.h"
```

Functions

- const BeidouLdpcGraph & **beidou_cnav2_ldpc_graph_96_48** ()
- bool **beidou_cnav2_ldpc_decode_96_48** (const float *bit_llr, int32_t num_bits, uint8_t *info_bits288)
- bool **beidou_cnav2_ldpc_decode_96_48_codeword** (const float *bit_llr, int32_t num_bits, uint8_↩ t *codeword_bits576)

Variables

- constexpr int32_t BEIDOU_CNAV2_LDPC_N = 96
- constexpr int32_t BEIDOU_CNAV2_LDPC_K = 48
- constexpr int32_t BEIDOU_CNAV2_LDPC_M = 48
- constexpr int32_t BEIDOU_CNAV2_LDPC_DC = 4
- constexpr uint16_t BEIDOU_CNAV2_H48_96_INDEX [192]
- constexpr uint8_t BEIDOU_CNAV2_H48_96_ELEMENT [192]

13.664.1 Detailed Description

B-CNAV2 LDPC(96,48) decoder (BDS-SIS-ICD-B2a-1.0 section 6.2.2).

Author

Carles Fernandez, 2026 cfernandez(at)cttc.cat

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_cnav2_ldpc.h.

13.665 beidou_cnav2_ldpc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav2_ldpc.h
00003  * \brief B-CNAV2 LDPC(96,48) decoder (BDS-SIS-ICD-B2a-1.0 section 6.2.2)
00004  * \author Carles Fernandez, 2026 cfernandez(at)cttc.cat
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_CNAV2_LDPC_H
00018 #define GNSS_SDR_BEIDOU_CNAV2_LDPC_H
00019
00020 #include "beidou_ldpc.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027 constexpr int32_t BEIDOU_CNAV2_LDPC_N = 96;
00028 constexpr int32_t BEIDOU_CNAV2_LDPC_K = 48;
00029 constexpr int32_t BEIDOU_CNAV2_LDPC_M = 48;
00030 constexpr int32_t BEIDOU_CNAV2_LDPC_DC = 4;
00031
00032 // ICD section 6.2.2, pp. 20-21: printed order, four 4-column bands.
00033 // beidou_ldpc_init_graph reads each band top to bottom before the next band.
00034 constexpr uint16_t BEIDOU_CNAV2_H48_96_INDEX[192] = {
00035     19, 46, 49, 76, 5, 29, 53, 71, 17, 30, 64, 72, 22, 36, 59, 82,
00036     22, 41, 68, 94, 20, 44, 54, 75, 9, 41, 61, 86, 6, 47, 60, 89,
00037     8, 40, 60, 87, 15, 26, 66, 81, 19, 24, 67, 95, 2, 26, 50, 72,
00038     5, 38, 70, 89, 16, 34, 64, 92, 21, 45, 55, 74, 0, 24, 48, 78,
00039     23, 37, 58, 83, 15, 43, 56, 91, 18, 47, 48, 77, 14, 42, 57, 90,
00040     6, 30, 54, 76, 14, 27, 67, 80, 17, 35, 65, 93, 7, 46, 61, 88,
00041     1, 25, 49, 79, 12, 45, 69, 79, 18, 25, 66, 94, 23, 40, 69, 95,
00042     8, 36, 51, 84, 3, 38, 56, 86, 0, 29, 62, 85, 2, 39, 57, 87,
00043     11, 33, 59, 81, 20, 43, 74, 93, 13, 32, 63, 91, 11, 35, 52, 83,
00044     16, 31, 65, 73, 4, 28, 52, 70, 1, 28, 63, 84, 12, 33, 62, 90,
00045     21, 42, 75, 92, 7, 31, 55, 77, 9, 37, 50, 85, 10, 34, 53, 82,
00046     4, 39, 71, 88, 13, 44, 68, 78, 3, 27, 51, 73, 10, 32, 58, 80};
00047
00048 constexpr uint8_t BEIDOU_CNAV2_H48_96_ELEMENT[192] = {
00049     1, 45, 15, 6, 1, 44, 53, 24, 45, 15, 6, 1, 30, 24, 1, 44,
00050     18, 15, 32, 61, 3, 55, 9, 34, 35, 31, 50, 44, 45, 15, 6, 1,
00051     24, 1, 44, 53, 30, 24, 1, 44, 32, 42, 47, 37, 6, 1, 45, 15,
00052     44, 53, 24, 1, 39, 36, 34, 33, 44, 53, 24, 1, 44, 53, 24, 1,
00053     45, 15, 6, 1, 6, 1, 45, 15, 24, 1, 44, 53, 9, 41, 57, 58,
```

```
00054     32, 61, 18, 40, 1, 45, 15, 6, 22, 14, 2, 50, 24, 1, 44, 30,
00055     30, 24, 1, 44, 15, 46, 45, 44, 45, 15, 6, 1, 1, 44, 30, 24,
00056     24, 1, 44, 53, 15, 6, 1, 45, 53, 24, 1, 44, 7, 38, 23, 54,
00057     1, 45, 15, 6, 44, 53, 24, 1, 57, 25, 9, 41, 35, 13, 51, 60,
00058     33, 45, 36, 34, 6, 1, 45, 15, 6, 1, 45, 15, 6, 1, 45, 15,
00059     44, 35, 31, 50, 26, 27, 37, 5, 24, 1, 44, 30, 33, 42, 14, 5,
00060     24, 1, 44, 30, 24, 1, 44, 30, 1, 44, 53, 24, 1, 44, 30, 24};
00061
00062 const BeidouLdpcGraph& beidou_cnav2_ldpc_graph_96_48();
00063
00064 // Input: 576 bit LLRs (positive favors 1); output: 288 MSB-first information bits.
00065 bool beidou_cnav2_ldpc_decode_96_48(const float* bit_llr, int32_t num_bits, uint8_t* info_bits288);
00066 bool beidou_cnav2_ldpc_decode_96_48_codeword(const float* bit_llr, int32_t num_bits, uint8_t*
      codeword_bits576);
00067
00068 /** \} */
00069 /** \} */
00070 #endif  // GNSS_SDR_BEIDOU_CNAV2_LDPC_H
```

13.666 beidou_cnav2_navigation_message.h File Reference

B-CNAV2 frame parser (BDS-SIS-ICD-B2a-1.0).

```
#include "beidou_cnav1_ephemeris.h"
#include <array>
#include <cstdint>
#include <string>
```

Classes

- class Beidou_Cnav2_Navigation_Message

13.666.1 Detailed Description

B-CNAV2 frame parser (BDS-SIS-ICD-B2a-1.0).

Author

huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_cnav2_navigation_message.h.

13.667 beidou_cnav2_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_cnav2_navigation_message.h
00003  * \brief B-CNAV2 frame parser (BDS-SIS-ICD-B2a-1.0).
00004  * \author huangchuhan, 2026. huangchh37(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_CNAV2_NAVIGATION_MESSAGE_H
00018 #define GNSS_SDR_BEIDOU_CNAV2_NAVIGATION_MESSAGE_H
00019
00020 #include "beidou_cnav1_ephemeris.h"
00021 #include <array>
00022 #include <cstdint>
00023 #include <string>
00024
00025 /** \addtogroup Core
00026  * \{ */
```

```
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030 class Beidou_Cnav2_Navigation_Message
00031 {
00032 public:
00033     bool decode_frame_symbols(const float* symbols, int32_t n_symbols, uint32_t prn);
00034     bool have_new_ephemeris() const { return d_flag_new_eph; }
00035     Beidou_Cnav1_Ephemeris get_ephemeris() const { return d_published; }
00036     void clear_flags() { d_flag_new_eph = false; }
00037     void reset();
00038     bool last_crc_ok() const { return d_last_crc_ok; }
00039     int32_t last_mes_type() const { return d_last_mes_type; }
00040     int32_t last_sow() const { return d_last_sow; }
00041     uint32_t last_frame_prn() const { return d_last_frame_prn; }
00042     const std::string& get_last_nav_bits() const { return d_last_nav_bits; }
00043
00044     /*!
00045      * \brief Soft preamble detection statistic on 1 ms prompt samples.
00046      *
00047      * current_1ms points to BEIDOU_CNAV2_PREAMBLE_MS samples at a candidate frame
00048      * start, previous_1ms (optional) to the samples one frame (3 s) earlier.
00049      * With C the correlation with the preamble spread by the data secondary code
00050      * and E the sample energy, it returns max(|C1| / sqrt(E1), |C1 + C2| / sqrt(E1 + E2)).
00051      * The result does not depend on the input scale and is about |N(0,1)| on
00052      * noise. The single-frame term keeps detection working across a carrier
00053      * polarity flip between the two frames.
00054      */
00055     static double preamble_detection_statistic(const float* current_1ms, const float* previous_1ms =
    nullptr);
00056
00057 private:
00058     void parse_info_bits(const uint8_t* bits, uint32_t channel_prn);
00059     void parse_clock_common(const uint8_t* bits, int32_t toc_off, int32_t iodc_off);
00060     void update_health(const uint8_t* bits);
00061     void try_publish();
00062     bool orbit_compatible() const;
00063     bool set_compatible() const;
00064
00065     Beidou_Cnav1_Ephemeris d_published{};
00066     Beidou_Cnav1_Ephemeris d_cand{};
00067     std::string d_last_nav_bits;
00068     int32_t d_last_mes_type{-1};
00069     int32_t d_last_sow{-1};
00070     int32_t d_iode_mt10{-1};
00071     int32_t d_iodc{-1};
00072     uint32_t d_last_frame_prn{};
00073     bool d_have_mt10{false};
00074     bool d_have_mt11{false};
00075     bool d_have_clk{false};
00076     bool d_have_published{false};
00077     bool d_flag_new_eph{false};
00078     bool d_last_crc_ok{false};
00079 };
00080
00081 /** \} */
00082 /** \} */
00083 #endif  // GNSS_SDR_BEIDOU_CNAV2_NAVIGATION_MESSAGE_H
```

13.668 Beidou_DNAV.h File Reference

Defines system parameters for BeiDou DNAV data processing.

```
#include "MATH_CONSTANTS.h"
#include <cstddef>
#include <cstdint>
#include <utility>
#include <vector>
```

Functions

- const std::vector< std::pair< int32_t, int32_t > > **D1_PRE** ({{1, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_FRAID** ({{16, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SOW** ({{19, 8}, {31, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_PNUM** ({{44, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SAT_H1** ({{43, 1}})

- const std::vector< std::pair< int32_t, int32_t > > **D1_AODC** ({{44, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_URAI** ({{49, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WN** ({{61, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOC** ({{74, 9}, {91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TGD1** ({{99, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TGD2** ({{109, 4}, {121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA0** ({{127, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA1** ({{135, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA2** ({{151, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_ALPHA3** ({{159, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA0** ({{167, 6}, {181, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA1** ({{183, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA2** ({{191, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_BETA3** ({{199, 4}, {211, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A2** ({{215, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0** ({{226, 7}, {241, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1** ({{258, 5}, {271, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AODE** ({{288, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_N** ({{43, 10}, {61, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CUC** ({{67, 16}, {91, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_M0** ({{93, 20}, {121, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_E** ({{133, 10}, {151, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CUS** ({{181, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CRC** ({{199, 4}, {211, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CRS** ({{225, 8}, {241, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SQRT_A** ({{251, 12}, {271, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOE_SF2** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOE_SF3** ({{43, 10}, {61, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_I0** ({{66, 17}, {91, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CIC** ({{106, 7}, {121, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_DOT** ({{132, 11}, {151, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_CIS** ({{164, 9}, {181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_IDOT** ({{190, 13}, {211, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA0** ({{212, 21}, {241, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA** ({{252, 11}, {271, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_SQRT_A_ALMANAC** ({{51, 2}, {61, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1_ALMANAC** ({{91, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0_ALMANAC** ({{102, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA0_ALMANAC** ({{121, 22}, {151, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_E_ALMANAC** ({{153, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_I** ({{170, 3}, {181, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOA** ({{194, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_DOT_ALMANAC** ({{202, 1}, {211, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_OMEGA_ALMANAC** ({{227, 6}, {241, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_M0_ALMANAC** ({{259, 4}, {271, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMEPID** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMID** ({{291, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_AMID_HEALTH** ({{216, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA1** ({{51, 2}, {61, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA2** ({{68, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA3** ({{77, 6}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA4** ({{94, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA5** ({{103, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA6** ({{112, 1}, {121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA7** ({{129, 9}})

- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA8** ({{138, 5}, {151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA9** ({{155, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA10** ({{164, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA11** ({{181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA12** ({{190, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA13** ({{199, 4}, {211, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA14** ({{216, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA15** ({{225, 8}, {241, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA16** ({{242, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA17** ({{251, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA18** ({{260, 3}, {271, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA19** ({{277, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA20** ({{51, 2}, {61, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA21** ({{68, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA22** ({{77, 6}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA23** ({{94, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA24** ({{103, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA25** ({{112, 1}, {121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA26** ({{129, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA27** ({{138, 5}, {151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA28** ({{155, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA29** ({{164, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_HEA30** ({{181, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WNA** ({{190, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_TOA2** ({{198, 5}, {211, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GPS** ({{97, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GPS** ({{111, 2}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GAL** ({{135, 8}, {151, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GAL** ({{157, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0GLO** ({{181, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1GLO** ({{195, 8}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_T_LS** ({{51, 2}, {61, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DELTA_T_LSF** ({{67, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_WN_LSF** ({{75, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A0UTC** ({{91, 22}, {121, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_A1UTC** ({{131, 12}, {151, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D1_DN** ({{163, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PRE** ({{1, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_FRAID** ({{16, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SOW** ({{19, 8}, {31, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM1** ({{43, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM** (D2_PNUM1)
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM2** ({{44, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_PNUM5** ({{44, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SAT_H1** ({{47, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_AODC** ({{48, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_URAI** ({{61, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_WN** ({{65, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TOC** ({{78, 5}, {91, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TGD1** ({{103, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TGD2** ({{121, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA0** ({{47, 6}, {61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA1** ({{63, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA2** ({{71, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_ALPHA3** ({{79, 4}, {91, 4}})

- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA0** ({{95, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA1** ({{103, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA2** ({{111, 2}, {121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BETA3** ({{127, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A0** ({{101, 12}, {121, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1_MSB** ({{133, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1_LSB** ({{47, 6}, {61, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A1** ({{279, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_A2** ({{73, 10}, {91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_AODE** ({{92, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_N** ({{97, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC_MSB** ({{121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC_LSB** ({{47, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUC** ({{283, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_M0** ({{51, 2}, {61, 22}, {91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CUS** ({{99, 14}, {121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_E_MSB** ({{125, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_E_LSB** ({{47, 6}, {61, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SQRT_A** ({{77, 6}, {91, 22}, {121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC_MSB** ({{125, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC_LSB** ({{47, 6}, {61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIC** ({{283, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CIS** ({{63, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_TOE** ({{81, 2}, {91, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0_MSB** ({{106, 7}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0_LSB** ({{47, 6}, {61, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_I0** ({{269, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CRC** ({{66, 17}, {91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_CRS** ({{92, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT_MSB** ({{110, 3}, {121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT_LSB** ({{47, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_DOT** ({{277, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA0** ({{52, 1}, {61, 22}, {91, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_MSB** ({{100, 13}, {121, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA_LSB** ({{47, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_OMEGA** ({{269, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_IDOT** ({{52, 1}, {61, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_SAT_H2** ({{48, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDID1_30** ({{50, 3}, {61, 22}, {91, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF2** ({{276, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF2** ({{280, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF3_1** ({{44, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF3_1** ({{48, 5}, {61, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF3_2** ({{69, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF3_2** ({{73, 10}, {91, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDEPID** ({{169, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_BDID31_63** ({{171, 2}, {181, 22}, {211, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_RURAI_SF4** ({{252, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **D2_DELTA_T_SF4** ({{256, 7}, {271, 6}})

Variables

- constexpr double D1_TOC_LSB = TWO_P3
- constexpr double D1_TGD1_LSB = 0.1e-9
- constexpr double D1_TGD2_LSB = 0.1e-9
- constexpr double D1_ALPHA0_LSB = TWO_N30
- constexpr double D1_ALPHA1_LSB = TWO_N27
- constexpr double D1_ALPHA2_LSB = TWO_N24
- constexpr double D1_ALPHA3_LSB = TWO_N24
- constexpr double D1_BETA0_LSB = TWO_P11
- constexpr double D1_BETA1_LSB = TWO_P14
- constexpr double D1_BETA2_LSB = TWO_P16
- constexpr double D1_BETA3_LSB = TWO_P16
- constexpr double D1_A2_LSB = TWO_N66
- constexpr double D1_A0_LSB = TWO_N33
- constexpr double D1_A1_LSB = TWO_N50
- constexpr double D1_DELTA_N_LSB = PI_TWO_N43
- constexpr double D1_CUC_LSB = TWO_N31
- constexpr double D1_M0_LSB = PI_TWO_N31
- constexpr double D1_E_LSB = TWO_N33
- constexpr double D1_CUS_LSB = TWO_N31
- constexpr double D1_CRC_LSB = TWO_N6
- constexpr double D1_CRS_LSB = TWO_N6
- constexpr double D1_SQRT_A_LSB = TWO_N19
- constexpr double D1_TOE_LSB = TWO_P3
- constexpr double D1_I0_LSB = PI_TWO_N31
- constexpr double D1_CIC_LSB = TWO_N31
- constexpr double D1_OMEGA_DOT_LSB = PI_TWO_N43
- constexpr double D1_CIS_LSB = TWO_N31
- constexpr double D1_IDOT_LSB = PI_TWO_N43
- constexpr double D1_OMEGA0_LSB = PI_TWO_N31
- constexpr double D1_OMEGA_LSB = PI_TWO_N31
- constexpr double D1_SQRT_A_ALMANAC_LSB = TWO_N11
- constexpr double D1_A1_ALMANAC_LSB = TWO_N38
- constexpr double D1_A0_ALMANAC_LSB = TWO_N20
- constexpr double D1_OMEGA0_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_E_ALMANAC_LSB = TWO_N21
- constexpr double D1_DELTA_I_LSB = PI_TWO_N19
- constexpr double D1_TOA_LSB = TWO_P12
- constexpr double D1_OMEGA_DOT_ALMANAC_LSB = PI_TWO_N38
- constexpr double D1_OMEGA_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_M0_ALMANAC_LSB = PI_TWO_N23
- constexpr double D1_A0GPS_LSB = 0.1e-9
- constexpr double D1_A1GPS_LSB = 0.1e-9
- constexpr double D1_A0GAL_LSB = 0.1e-9
- constexpr double D1_A1GAL_LSB = 0.1e-9
- constexpr double D1_A0GLO_LSB = 0.1e-9
- constexpr double D1_A1GLO_LSB = 0.1e-9
- constexpr double D1_A0UTC_LSB = TWO_N30
- constexpr double D1_A1UTC_LSB = TWO_N50
- constexpr double D2_DELTA_T_LSB = 0.1
- constexpr double D2_ION_DELAY_LSB = 0.125
- constexpr std::size_t BEIDOU_DNAV_BCH_CODE_BITS = 15
- constexpr std::size_t BEIDOU_DNAV_BCH_DATA_BITS = 11
- constexpr std::size_t BEIDOU_DNAV_BCH_PARITY_BITS = 4

- constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_BITS = 11
- constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS = 11
- constexpr int32_t BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS = 300
- constexpr int32_t BEIDOU_DNAV_SUBFRAME_DATA_BITS = 300

  Number of bits per subframe in the NAV message [bits].
- constexpr double BEIDOU_DNAV_SECONDS_PER_WEEK = 604800.0
- constexpr double BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS = 6.0
- constexpr double BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS = 3.0
- constexpr int32_t BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET = 14
- constexpr int32_t BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET = 1356
- constexpr uint32_t BEIDOU_DNAV_SUBFRAME_SYMBOLS = 300
- constexpr uint32_t BEIDOU_DNAV_WORDS_SUBFRAME = 10
- constexpr uint32_t BEIDOU_DNAV_WORD_LENGTH_BITS = 30
- constexpr char BEIDOU_DNAV_PREAMBLE [12] = "11100010010"
- constexpr int32_t D2_UDREI1_LOGICAL_FIRST_BIT = 172
- constexpr int32_t D2_UDREI19_LOGICAL_FIRST_BIT = 220
- constexpr int32_t D2_ION_LOGICAL_FIRST_BIT = 51

13.668.1 Detailed Description

Defines system parameters for BeiDou DNAV data processing.

Author

Damian Miralles, 2018. dmiralles2009@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file Beidou_DNAV.h.

13.669 Beidou_DNAV.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Beidou_DNAV.h
00003  * \brief  Defines system parameters for BeiDou DNAV data processing
00004  * \author Damian Miralles, 2018. dmiralles2009@gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_BEIDOU_DNAV_H
00019 #define GNSS_SDR_BEIDOU_DNAV_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include <cstddef>
00023 #include <cstdint>
00024 #include <utility>
00025 #include <vector>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 // DNAV SCALE FACTORS
00034 // EPH
00035 constexpr double D1_TOC_LSB = TWO_P3;
00036 constexpr double D1_TGD1_LSB = 0.1e-9;
00037 constexpr double D1_TGD2_LSB = 0.1e-9;
```

```
00038 constexpr double D1_ALPHA0_LSB = TWO_N30;
00039 constexpr double D1_ALPHA1_LSB = TWO_N27;
00040 constexpr double D1_ALPHA2_LSB = TWO_N24;
00041 constexpr double D1_ALPHA3_LSB = TWO_N24;
00042 constexpr double D1_BETA0_LSB = TWO_P11;
00043 constexpr double D1_BETA1_LSB = TWO_P14;
00044 constexpr double D1_BETA2_LSB = TWO_P16;
00045 constexpr double D1_BETA3_LSB = TWO_P16;
00046 constexpr double D1_A2_LSB = TWO_N66;
00047 constexpr double D1_A0_LSB = TWO_N33;
00048 constexpr double D1_A1_LSB = TWO_N50;
00049 constexpr double D1_DELTA_N_LSB = PI_TWO_N43;
00050 constexpr double D1_CUC_LSB = TWO_N31;
00051 constexpr double D1_M0_LSB = PI_TWO_N31;
00052 constexpr double D1_E_LSB = TWO_N33;
00053 constexpr double D1_CUS_LSB = TWO_N31;
00054 constexpr double D1_CRC_LSB = TWO_N6;
00055 constexpr double D1_CRS_LSB = TWO_N6;
00056 constexpr double D1_SQRT_A_LSB = TWO_N19;
00057 constexpr double D1_TOE_LSB = TWO_P3;
00058 constexpr double D1_I0_LSB = PI_TWO_N31;
00059 constexpr double D1_CIC_LSB = TWO_N31;
00060 constexpr double D1_OMEGA_DOT_LSB = PI_TWO_N43;
00061 constexpr double D1_CIS_LSB = TWO_N31;
00062 constexpr double D1_IDOT_LSB = PI_TWO_N43;
00063 constexpr double D1_OMEGA0_LSB = PI_TWO_N31;
00064 constexpr double D1_OMEGA_LSB = PI_TWO_N31;
00065 // ALM
00066 constexpr double D1_SQRT_A_ALMANAC_LSB = TWO_N11;
00067 constexpr double D1_A1_ALMANAC_LSB = TWO_N38;
00068 constexpr double D1_A0_ALMANAC_LSB = TWO_N20;
00069 constexpr double D1_OMEGA0_ALMANAC_LSB = PI_TWO_N23;
00070 constexpr double D1_E_ALMANAC_LSB = TWO_N21;
00071 constexpr double D1_DELTA_I_LSB = PI_TWO_N19;
00072 constexpr double D1_TOA_LSB = TWO_P12;
00073 constexpr double D1_OMEGA_DOT_ALMANAC_LSB = PI_TWO_N38;
00074 constexpr double D1_OMEGA_ALMANAC_LSB = PI_TWO_N23;
00075 constexpr double D1_M0_ALMANAC_LSB = PI_TWO_N23;
00076 constexpr double D1_A0GPS_LSB = 0.1e-9;
00077 constexpr double D1_A1GPS_LSB = 0.1e-9;
00078 constexpr double D1_A0GAL_LSB = 0.1e-9;
00079 constexpr double D1_A1GAL_LSB = 0.1e-9;
00080 constexpr double D1_A0GLO_LSB = 0.1e-9;
00081 constexpr double D1_A1GLO_LSB = 0.1e-9;
00082 constexpr double D1_A0UTC_LSB = TWO_N30;
00083 constexpr double D1_A1UTC_LSB = TWO_N50;
00084 constexpr double D2_DELTA_T_LSB = 0.1;  // Equivalent clock correction [m]
00085 constexpr double D2_ION_DELAY_LSB = 0.125;
00086
00087 constexpr std::size_t BEIDOU_DNAV_BCH_CODE_BITS = 15;
00088 constexpr std::size_t BEIDOU_DNAV_BCH_DATA_BITS = 11;
00089 constexpr std::size_t BEIDOU_DNAV_BCH_PARITY_BITS = 4;
00090
00091 constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_BITS = 11;
00092 constexpr int32_t BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS = 11;  // **************
00093 constexpr int32_t BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS = 300;
00094 constexpr int32_t BEIDOU_DNAV_SUBFRAME_DATA_BITS = 300;  //!< Number of bits per subframe in the NAV
     message [bits]
00095 constexpr double BEIDOU_DNAV_SECONDS_PER_WEEK = 604800.0;
00096 constexpr double BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS = 6.0;
00097 constexpr double BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS = 3.0;
00098 // Number of leap seconds passed from the start of the GPS epoch up to the start of BeiDou epoch
00099 constexpr int32_t BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET = 14;
00100 // Number of weeks passed from the start of the GPS epoch up to the start of BeiDou epoch
00101 constexpr int32_t BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET = 1356;
00102
00103 constexpr uint32_t BEIDOU_DNAV_SUBFRAME_SYMBOLS = 300;
00104 constexpr uint32_t BEIDOU_DNAV_WORDS_SUBFRAME = 10;
00105 constexpr uint32_t BEIDOU_DNAV_WORD_LENGTH_BITS = 30;
00106 constexpr char BEIDOU_DNAV_PREAMBLE[12] = "11100010010";
00107
00108
00109 // BEIDOU D1 NAVIGATION MESSAGE STRUCTURE
00110 // GENERAL
00111 const std::vector<std::pair<int32_t, int32_t> > D1_PRE({{1, 11}});
00112 const std::vector<std::pair<int32_t, int32_t> > D1_FRAID({{16, 3}});
00113 const std::vector<std::pair<int32_t, int32_t> > D1_SOW({{19, 8}, {31, 12}});
00114 const std::vector<std::pair<int32_t, int32_t> > D1_PNUM({{44, 7}});
00115
00116 // SUBFRAME 1
00117 const std::vector<std::pair<int32_t, int32_t> > D1_SAT_H1({{43, 1}});
00118 const std::vector<std::pair<int32_t, int32_t> > D1_AODC({{44, 5}});
00119 const std::vector<std::pair<int32_t, int32_t> > D1_URAI({{49, 4}});
00120 const std::vector<std::pair<int32_t, int32_t> > D1_WN({{61, 13}});
00121 const std::vector<std::pair<int32_t, int32_t> > D1_TOC({{74, 9}, {91, 8}});
00122 const std::vector<std::pair<int32_t, int32_t> > D1_TGD1({{99, 10}});
00123 const std::vector<std::pair<int32_t, int32_t> > D1_TGD2({{109, 4}, {121, 6}});
```

```
00124 const std::vector<std::pair<int32_t, int32_t> > D1_ALPHA0({{127, 8}});
00125 const std::vector<std::pair<int32_t, int32_t> > D1_ALPHA1({{135, 8}});
00126 const std::vector<std::pair<int32_t, int32_t> > D1_ALPHA2({{151, 8}});
00127 const std::vector<std::pair<int32_t, int32_t> > D1_ALPHA3({{159, 8}});
00128 const std::vector<std::pair<int32_t, int32_t> > D1_BETA0({{167, 6}, {181, 2}});
00129 const std::vector<std::pair<int32_t, int32_t> > D1_BETA1({{183, 8}});
00130 const std::vector<std::pair<int32_t, int32_t> > D1_BETA2({{191, 8}});
00131 const std::vector<std::pair<int32_t, int32_t> > D1_BETA3({{199, 4}, {211, 4}});
00132 const std::vector<std::pair<int32_t, int32_t> > D1_A2({{215, 11}});
00133 const std::vector<std::pair<int32_t, int32_t> > D1_A0({{226, 7}, {241, 17}});
00134 const std::vector<std::pair<int32_t, int32_t> > D1_A1({{258, 5}, {271, 17}});
00135 const std::vector<std::pair<int32_t, int32_t> > D1_AODE({{288, 5}});
00136
00137 // SUBFRAME 2
00138 const std::vector<std::pair<int32_t, int32_t> > D1_DELTA_N({{43, 10}, {61, 6}});
00139 const std::vector<std::pair<int32_t, int32_t> > D1_CUC({{67, 16}, {91, 2}});
00140 const std::vector<std::pair<int32_t, int32_t> > D1_M0({{93, 20}, {121, 12}});
00141 const std::vector<std::pair<int32_t, int32_t> > D1_E({{133, 10}, {151, 22}});
00142 const std::vector<std::pair<int32_t, int32_t> > D1_CUS({{181, 18}});
00143 const std::vector<std::pair<int32_t, int32_t> > D1_CRC({{199, 4}, {211, 14}});
00144 const std::vector<std::pair<int32_t, int32_t> > D1_CRS({{225, 8}, {241, 10}});
00145 const std::vector<std::pair<int32_t, int32_t> > D1_SQRT_A({{251, 12}, {271, 20}});
00146 const std::vector<std::pair<int32_t, int32_t> > D1_TOE_SF2({{291, 2}});
00147
00148 // SUBFRAME 3
00149 const std::vector<std::pair<int32_t, int32_t> > D1_TOE_SF3({{43, 10}, {61, 5}});
00150 const std::vector<std::pair<int32_t, int32_t> > D1_I0({{66, 17}, {91, 15}});
00151 const std::vector<std::pair<int32_t, int32_t> > D1_CIC({{106, 7}, {121, 11}});
00152 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA_DOT({{132, 11}, {151, 13}});
00153 const std::vector<std::pair<int32_t, int32_t> > D1_CIS({{164, 9}, {181, 9}});
00154 const std::vector<std::pair<int32_t, int32_t> > D1_IDOT({{190, 13}, {211, 1}});
00155 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA0({{212, 21}, {241, 11}});
00156 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA({{252, 11}, {271, 21}});
00157
00158 // SUBFRAME 4 AND PAGES 1 THROUGH 6 IN SUBFRAME 5
00159 const std::vector<std::pair<int32_t, int32_t> > D1_SQRT_A_ALMANAC({{51, 2}, {61, 22}});
00160 const std::vector<std::pair<int32_t, int32_t> > D1_A1_ALMANAC({{91, 11}});
00161 const std::vector<std::pair<int32_t, int32_t> > D1_A0_ALMANAC({{102, 11}});
00162 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA0_ALMANAC({{121, 22}, {151, 2}});
00163 const std::vector<std::pair<int32_t, int32_t> > D1_E_ALMANAC({{153, 17}});
00164 const std::vector<std::pair<int32_t, int32_t> > D1_DELTA_I({{170, 3}, {181, 13}});
00165 const std::vector<std::pair<int32_t, int32_t> > D1_TOA({{194, 8}});
00166 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA_DOT_ALMANAC({{202, 1}, {211, 16}});
00167 const std::vector<std::pair<int32_t, int32_t> > D1_OMEGA_ALMANAC({{227, 6}, {241, 18}});
00168 const std::vector<std::pair<int32_t, int32_t> > D1_M0_ALMANAC({{259, 4}, {271, 20}});
00169 const std::vector<std::pair<int32_t, int32_t> > D1_AMEPID({{291, 2}});
00170 const std::vector<std::pair<int32_t, int32_t> > D1_AMID({{291, 2}});
00171 const std::vector<std::pair<int32_t, int32_t> > D1_AMID_HEALTH({{216, 2}});
00172
00173 // SUBFRAME 5 PAGE 7
00174 const std::vector<std::pair<int32_t, int32_t> > D1_HEA1({{51, 2}, {61, 7}});
00175 const std::vector<std::pair<int32_t, int32_t> > D1_HEA2({{68, 9}});
00176 const std::vector<std::pair<int32_t, int32_t> > D1_HEA3({{77, 6}, {91, 3}});
00177 const std::vector<std::pair<int32_t, int32_t> > D1_HEA4({{94, 9}});
00178 const std::vector<std::pair<int32_t, int32_t> > D1_HEA5({{103, 9}});
00179 const std::vector<std::pair<int32_t, int32_t> > D1_HEA6({{112, 1}, {121, 8}});
00180 const std::vector<std::pair<int32_t, int32_t> > D1_HEA7({{129, 9}});
00181 const std::vector<std::pair<int32_t, int32_t> > D1_HEA8({{138, 5}, {151, 4}});
00182 const std::vector<std::pair<int32_t, int32_t> > D1_HEA9({{155, 9}});
00183 const std::vector<std::pair<int32_t, int32_t> > D1_HEA10({{164, 9}});
00184 const std::vector<std::pair<int32_t, int32_t> > D1_HEA11({{181, 9}});
00185 const std::vector<std::pair<int32_t, int32_t> > D1_HEA12({{190, 9}});
00186 const std::vector<std::pair<int32_t, int32_t> > D1_HEA13({{199, 4}, {211, 5}});
00187 const std::vector<std::pair<int32_t, int32_t> > D1_HEA14({{216, 9}});
00188 const std::vector<std::pair<int32_t, int32_t> > D1_HEA15({{225, 8}, {241, 1}});
00189 const std::vector<std::pair<int32_t, int32_t> > D1_HEA16({{242, 9}});
00190 const std::vector<std::pair<int32_t, int32_t> > D1_HEA17({{251, 9}});
00191 const std::vector<std::pair<int32_t, int32_t> > D1_HEA18({{260, 3}, {271, 6}});
00192 const std::vector<std::pair<int32_t, int32_t> > D1_HEA19({{277, 9}});
00193
00194 // SUBFRAME 5 PAGE 8
00195 const std::vector<std::pair<int32_t, int32_t> > D1_HEA20({{51, 2}, {61, 7}});
00196 const std::vector<std::pair<int32_t, int32_t> > D1_HEA21({{68, 9}});
00197 const std::vector<std::pair<int32_t, int32_t> > D1_HEA22({{77, 6}, {91, 3}});
00198 const std::vector<std::pair<int32_t, int32_t> > D1_HEA23({{94, 9}});
00199 const std::vector<std::pair<int32_t, int32_t> > D1_HEA24({{103, 9}});
00200 const std::vector<std::pair<int32_t, int32_t> > D1_HEA25({{112, 1}, {121, 8}});
00201 const std::vector<std::pair<int32_t, int32_t> > D1_HEA26({{129, 9}});
00202 const std::vector<std::pair<int32_t, int32_t> > D1_HEA27({{138, 5}, {151, 4}});
00203 const std::vector<std::pair<int32_t, int32_t> > D1_HEA28({{155, 9}});
00204 const std::vector<std::pair<int32_t, int32_t> > D1_HEA29({{164, 9}});
00205 const std::vector<std::pair<int32_t, int32_t> > D1_HEA30({{181, 9}});
00206 const std::vector<std::pair<int32_t, int32_t> > D1_WNA({{190, 8}});
00207 const std::vector<std::pair<int32_t, int32_t> > D1_TOA2({{198, 5}, {211, 3}});
00208
00209 // SUBFRAME 5 PAGE 9
00210 const std::vector<std::pair<int32_t, int32_t> > D1_A0GPS({{97, 14}});
```

```
00211 const std::vector<std::pair<int32_t, int32_t> > D1_A1GPS({{111, 2}, {121, 14}});
00212 const std::vector<std::pair<int32_t, int32_t> > D1_A0GAL({{135, 8}, {151, 6}});
00213 const std::vector<std::pair<int32_t, int32_t> > D1_A1GAL({{157, 16}});
00214 const std::vector<std::pair<int32_t, int32_t> > D1_A0GLO({{181, 14}});
00215 const std::vector<std::pair<int32_t, int32_t> > D1_A1GLO({{195, 8}, {211, 8}});
00216
00217 // SUBFRAME 5 PAGE 10
00218 const std::vector<std::pair<int32_t, int32_t> > D1_DELTA_T_LS({{51, 2}, {61, 6}});
00219 const std::vector<std::pair<int32_t, int32_t> > D1_DELTA_T_LSF({{67, 8}});
00220 const std::vector<std::pair<int32_t, int32_t> > D1_WN_LSF({{75, 8}});
00221 const std::vector<std::pair<int32_t, int32_t> > D1_A0UTC({{91, 22}, {121, 10}});
00222 const std::vector<std::pair<int32_t, int32_t> > D1_A1UTC({{131, 12}, {151, 12}});
00223 const std::vector<std::pair<int32_t, int32_t> > D1_DN({{163, 8}});
00224
00225 // D2 NAV Message Decoding Information
00226 const std::vector<std::pair<int32_t, int32_t> > D2_PRE({{1, 11}});
00227 const std::vector<std::pair<int32_t, int32_t> > D2_FRAID({{16, 3}});
00228 const std::vector<std::pair<int32_t, int32_t> > D2_SOW({{19, 8}, {31, 12}});
00229 const std::vector<std::pair<int32_t, int32_t> > D2_PNUM1({{43, 4}});
00230 const std::vector<std::pair<int32_t, int32_t> > D2_PNUM(D2_PNUM1);  // Backward-compatible name
00231 const std::vector<std::pair<int32_t, int32_t> > D2_PNUM2({{44, 4}});
00232 const std::vector<std::pair<int32_t, int32_t> > D2_PNUM5({{44, 7}});
00233
00234 // D2 NAV, SUBFRAME 1, PAGE 1
00235 const std::vector<std::pair<int32_t, int32_t> > D2_SAT_H1({{47, 1}});
00236 const std::vector<std::pair<int32_t, int32_t> > D2_AODC({{48, 5}});
00237 const std::vector<std::pair<int32_t, int32_t> > D2_URAI({{61, 4}});
00238 const std::vector<std::pair<int32_t, int32_t> > D2_WN({{65, 13}});
00239 const std::vector<std::pair<int32_t, int32_t> > D2_TOC({{78, 5}, {91, 12}});
00240 const std::vector<std::pair<int32_t, int32_t> > D2_TGD1({{103, 10}});
00241 const std::vector<std::pair<int32_t, int32_t> > D2_TGD2({{121, 10}});
00242
00243 // D2 NAV, SUBFRAME 1, PAGE 2
00244 const std::vector<std::pair<int32_t, int32_t> > D2_ALPHA0({{47, 6}, {61, 2}});
00245 const std::vector<std::pair<int32_t, int32_t> > D2_ALPHA1({{63, 8}});
00246 const std::vector<std::pair<int32_t, int32_t> > D2_ALPHA2({{71, 8}});
00247 const std::vector<std::pair<int32_t, int32_t> > D2_ALPHA3({{79, 4}, {91, 4}});
00248 const std::vector<std::pair<int32_t, int32_t> > D2_BETA0({{95, 8}});
00249 const std::vector<std::pair<int32_t, int32_t> > D2_BETA1({{103, 8}});
00250 const std::vector<std::pair<int32_t, int32_t> > D2_BETA2({{111, 2}, {121, 6}});
00251 const std::vector<std::pair<int32_t, int32_t> > D2_BETA3({{127, 8}});
00252
00253 // D2 NAV, SUBFRAME 1, PAGE 3
00254 const std::vector<std::pair<int32_t, int32_t> > D2_A0({{101, 12}, {121, 12}});
00255 const std::vector<std::pair<int32_t, int32_t> > D2_A1_MSB({{133, 4}});
00256 const std::vector<std::pair<int32_t, int32_t> > D2_A1_LSB({{47, 6}, {61, 12}});
00257 const std::vector<std::pair<int32_t, int32_t> > D2_A1({{279, 22}});
00258
00259 // D2 NAV, SUBFRAME 1, PAGE 4
00260 const std::vector<std::pair<int32_t, int32_t> > D2_A2({{73, 10}, {91, 1}});
00261 const std::vector<std::pair<int32_t, int32_t> > D2_AODE({{92, 5}});
00262 const std::vector<std::pair<int32_t, int32_t> > D2_DELTA_N({{97, 16}});
00263 const std::vector<std::pair<int32_t, int32_t> > D2_CUC_MSB({{121, 14}});
00264 const std::vector<std::pair<int32_t, int32_t> > D2_CUC_LSB({{47, 4}});
00265 const std::vector<std::pair<int32_t, int32_t> > D2_CUC({{283, 18}});
00266
00267 // D2 NAV, SUBFRAME 1, PAGE 5
00268 const std::vector<std::pair<int32_t, int32_t> > D2_M0({{51, 2}, {61, 22}, {91, 8}});
00269 const std::vector<std::pair<int32_t, int32_t> > D2_CUS({{99, 14}, {121, 4}});
00270 const std::vector<std::pair<int32_t, int32_t> > D2_E_MSB({{125, 10}});
00271
00272 // D2 NAV, SUBFRAME 1, PAGE 6
00273 const std::vector<std::pair<int32_t, int32_t> > D2_E_LSB({{47, 6}, {61, 16}});
00274 const std::vector<std::pair<int32_t, int32_t> > D2_SQRT_A({{77, 6}, {91, 22}, {121, 4}});
00275 const std::vector<std::pair<int32_t, int32_t> > D2_CIC_MSB({{125, 10}});
00276 const std::vector<std::pair<int32_t, int32_t> > D2_CIC_LSB({{47, 6}, {61, 2}});
00277 const std::vector<std::pair<int32_t, int32_t> > D2_CIC({{283, 18}});
00278
00279 // D2 NAV, SUBFRAME 1, PAGE 7
00280 const std::vector<std::pair<int32_t, int32_t> > D2_CIS({{63, 18}});
00281 const std::vector<std::pair<int32_t, int32_t> > D2_TOE({{81, 2}, {91, 15}});
00282 const std::vector<std::pair<int32_t, int32_t> > D2_I0_MSB({{106, 7}, {121, 14}});
00283 const std::vector<std::pair<int32_t, int32_t> > D2_I0_LSB({{47, 6}, {61, 5}});
00284 const std::vector<std::pair<int32_t, int32_t> > D2_I0({{269, 32}});
00285
00286 // D2 NAV, SUBFRAME 1, PAGE 8
00287 const std::vector<std::pair<int32_t, int32_t> > D2_CRC({{66, 17}, {91, 1}});
00288 const std::vector<std::pair<int32_t, int32_t> > D2_CRS({{92, 18}});
00289 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA_DOT_MSB({{110, 3}, {121, 16}});
00290 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA_DOT_LSB({{47, 5}});
00291 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA_DOT({{277, 24}});
00292
00293 // D2 NAV, SUBFRAME 1, PAGE 9
00294 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA0({{52, 1}, {61, 22}, {91, 9}});
00295 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA_MSB({{100, 13}, {121, 14}});
00296 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA_LSB({{47, 5}});
00297 const std::vector<std::pair<int32_t, int32_t> > D2_OMEGA({{269, 32}});
```

```
00298
00299 // D2 NAV, SUBFRAME 1, PAGE 10
00300 const std::vector<std::pair<int32_t, int32_t> > D2_IDOT({{52, 1}, {61, 13}});
00301
00302 // D2 NAV, SUBFRAME 2, PAGES 1 THROUGH 6
00303 const std::vector<std::pair<int32_t, int32_t> > D2_SAT_H2({{48, 2}});
00304 const std::vector<std::pair<int32_t, int32_t> > D2_BDID1_30({{50, 3}, {61, 22}, {91, 5}});
00305 const std::vector<std::pair<int32_t, int32_t> > D2_RURAI_SF2({{276, 4}});
00306 const std::vector<std::pair<int32_t, int32_t> > D2_DELTA_T_SF2({{280, 13}});
00307
00308 // D2 NAV, SUBFRAME 3, PAGES 1 THROUGH 6. The page number is carried by
00309 // subframe 2 in the same frame.
00310 const std::vector<std::pair<int32_t, int32_t> > D2_RURAI_SF3_1({{44, 4}});
00311 const std::vector<std::pair<int32_t, int32_t> > D2_DELTA_T_SF3_1({{48, 5}, {61, 8}});
00312 const std::vector<std::pair<int32_t, int32_t> > D2_RURAI_SF3_2({{69, 4}});
00313 const std::vector<std::pair<int32_t, int32_t> > D2_DELTA_T_SF3_2({{73, 10}, {91, 3}});
00314
00315 // D2 NAV, SUBFRAME 4, PAGES 1 THROUGH 6. Bits 171 through 300 are usable
00316 // only when BDEpID is 3.
00317 const std::vector<std::pair<int32_t, int32_t> > D2_BDEPID({{169, 2}});
00318 const std::vector<std::pair<int32_t, int32_t> > D2_BDID31_63({{171, 2}, {181, 22}, {211, 9}});
00319 const std::vector<std::pair<int32_t, int32_t> > D2_RURAI_SF4({{252, 4}});
00320 const std::vector<std::pair<int32_t, int32_t> > D2_DELTA_T_SF4({{256, 7}, {271, 6}});
00321
00322 // Logical payload offsets used for repeated D2 data. The first logical bit is
00323 // at the indicated transmitted bit and subsequent logical bits skip the BCH
00324 // parity positions.
00325 constexpr int32_t D2_UDREI1_LOGICAL_FIRST_BIT = 172;
00326 constexpr int32_t D2_UDREI19_LOGICAL_FIRST_BIT = 220;
00327 constexpr int32_t D2_ION_LOGICAL_FIRST_BIT = 51;
00328
00329
00330 /** \} */
00331 /** \} */
00332 #endif  // GNSS_SDR_BEIDOU_DNAV_H
```

13.670 beidou_dnav_almanac.h File Reference

Interface of a Beidou DNAV Almanac storage.

```
#include "gnss_almanac.h"
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
```

Classes

- class Beidou_Dnav_Almanac

  This class is a storage for the BeiDou D1/D2 almanac.

13.670.1 Detailed Description

Interface of a Beidou DNAV Almanac storage.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_dnav_almanac.h.

13.671 beidou_dnav_almanac.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_almanac.h
00003  * \brief  Interface of a Beidou DNAV Almanac storage
00004  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
```

```
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_BEIDOU_DNAV_ALMANAC_H
00019 #define GNSS_SDR_BEIDOU_DNAV_ALMANAC_H
00020
00021 #include "gnss_almanac.h"
00022 #include <boost/serialization/nvp.hpp>
00023 #include <boost/serialization/version.hpp>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief This class is a storage for the BeiDou D1/D2 almanac
00033  */
00034 class Beidou_Dnav_Almanac : public Gnss_Almanac
00035 {
00036 public:
00037     /*!
00038      * Default constructor
00039      */
00040     Beidou_Dnav_Almanac()
00041     {
00042         this->System = 'C';
00043     };
00044
00045     int SV_health{};  //!< SV Health
00046     int AmEpID{};     //!< Identification of the expanded-almanac sequence
00047     int AmID{};       //!< Identification of the expanded-almanac time-sharing group
00048     bool expanded{};  //!< True for almanacs of SV ID 31 through 63
00049
00050     template <class Archive>
00051
00052     void serialize(Archive& ar, const unsigned int version)
00053     {
00054         if (version)
00055             {
00056             };
00057         ar& BOOST_SERIALIZATION_NVP(PRN);
00058         ar& BOOST_SERIALIZATION_NVP(delta_i);
00059         ar& BOOST_SERIALIZATION_NVP(toa);
00060         ar& BOOST_SERIALIZATION_NVP(WNa);
00061         ar& BOOST_SERIALIZATION_NVP(M_0);
00062         ar& BOOST_SERIALIZATION_NVP(ecc);
00063         ar& BOOST_SERIALIZATION_NVP(sqrtA);
00064         ar& BOOST_SERIALIZATION_NVP(OMEGA_0);
00065         ar& BOOST_SERIALIZATION_NVP(omega);
00066         ar& BOOST_SERIALIZATION_NVP(OMEGAdot);
00067         ar& BOOST_SERIALIZATION_NVP(af0);
00068         ar& BOOST_SERIALIZATION_NVP(af1);
00069         ar& BOOST_SERIALIZATION_NVP(SV_health);
00070         if (version > 0)
00071             {
00072                 ar& BOOST_SERIALIZATION_NVP(AmEpID);
00073                 ar& BOOST_SERIALIZATION_NVP(AmID);
00074                 ar& BOOST_SERIALIZATION_NVP(expanded);
00075             }
00076     }
00077 };
00078
00079 BOOST_CLASS_VERSION(Beidou_Dnav_Almanac, 1)
00080
00081
00082 /** \} */
00083 /** \} */
00084 #endif  // GNSS_SDR_BEIDOU_DNAV_ALMANAC_H
```

13.672 beidou_dnav_differential_corrections.h File Reference

Storage for BeiDou D2 integrity and differential corrections.

```
#include <cstdint>
#include <map>
```

Classes

- struct Beidou_Dnav_Differential_Correction
- class Beidou_Dnav_Differential_Corrections

13.672.1 Detailed Description

Storage for BeiDou D2 integrity and differential corrections.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_dnav_differential_corrections.h.

13.673 beidou_dnav_differential_corrections.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_differential_corrections.h
00003  * \brief Storage for BeiDou D2 integrity and differential corrections
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_DNAV_DIFFERENTIAL_CORRECTIONS_H
00018 #define GNSS_SDR_BEIDOU_DNAV_DIFFERENTIAL_CORRECTIONS_H
00019
00020 #include <cstdint>
00021 #include <map>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028 struct Beidou_Dnav_Differential_Correction
00029 {
00030     uint32_t PRN{};
00031     int RURAI{};               //!< Regional User Range Accuracy Index
00032     int UDREI{};               //!< User Differential Range Error Index
00033     double delta_t{};          //!< Equivalent clock correction [m]
00034     bool delta_t_available{};  //!< False when the 13-bit value is -4096
00035 };
00036
00037
00038 class Beidou_Dnav_Differential_Corrections
00039 {
00040 public:
00041     uint32_t source_PRN{};  //!< GEO satellite broadcasting this data set
00042     double SOW{};
00043     int Pnum2{};
00044     int SatH2{};
00045     int BDEpID{};
00046     bool range_corrections_valid{};
00047     bool ionospheric_grid_valid{};
00048     bool expanded_corrections{};
00049     std::map<int, Beidou_Dnav_Differential_Correction> corrections;  //!< Keyed by corrected SV ID
00050 };
00051
00052 /** \} */
00053 /** \} */
00054 #endif  // GNSS_SDR_BEIDOU_DNAV_DIFFERENTIAL_CORRECTIONS_H
```

13.674 beidou_dnav_ephemeris.h File Reference

Interface of a BEIDOU EPHEMERIS storage.

```
#include "gnss_ephemeris.h"
#include <boost/serialization/nvp.hpp>
#include <map>
#include <string>
```

Classes

- class Beidou_Dnav_Ephemeris

  This is a storage class for the Beidou SV ephemeris data as described in BeiDou Navigation Satellite System Signal In Space Interface Control Document Open Service Signal B1I (Version 3.0).

13.674.1 Detailed Description

Interface of a BEIDOU EPHEMERIS storage.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_dnav_ephemeris.h.

13.675 beidou_dnav_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_ephemeris.h
00003  * \brief  Interface of a BEIDOU EPHEMERIS storage
00004  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_BEIDOU_DNAV_EPHEMERIS_H
00019 #define GNSS_SDR_BEIDOU_DNAV_EPHEMERIS_H
00020
00021 #include "gnss_ephemeris.h"
00022 #include <boost/serialization/nvp.hpp>
00023 #include <map>
00024 #include <string>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This is a storage class for the Beidou SV ephemeris data as described in
00034  * BeiDou Navigation Satellite System Signal In Space Interface Control Document
00035  * Open Service Signal B1I (Version 3.0)
00036  *
00037  * See http://en.beidou.gov.cn/SYSTEMS/Officialdocument/201902/P020190227601370045731.pdf
00038  */
00039 class Beidou_Dnav_Ephemeris : public Gnss_Ephemeris
00040 {
00041 public:
00042     /*!
00043      * Default constructor
00044      */
```

```
00045     Beidou_Dnav_Ephemeris();
00046
00047     int SV_accuracy{};  //!< User Range Accuracy (URA) index of the SV (reference paragraph 5.2.4.5)
      for the standard positioning service user
00048     int SV_health{};
00049     double TGD1{};  //!< Estimated Group Delay Differential on B1I [s]
00050     double TGD2{};  //!< Estimated Group Delay Differential on B2I [s]
00051     double AODC{};  //!< Age of Data, Clock
00052     double AODE{};  //!< Age of Data, Ephemeris
00053     int AODO{};     //!< Age of Data Offset (AODO) term for the navigation message correction table
      (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s]
00054
00055     int sig_type{};  //!< BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q) */
00056     int nav_type{};  //!< BDS: nav type (0:unknown,1:IGSO/MEO,2:GEO) */
00057
00058     bool fit_interval_flag{};  //!< Curve-fit interval used by the CS (Block II/IIA/IIR/IIR-M/IIF) and
      SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4 hours, 1 = greater than 4
      hours.
00059     double spare1{};
00060     double spare2{};
00061
00062     /*! \brief If true, enhanced level of integrity assurance.
00063      *
00064      *  If false, indicates that the conveying signal is provided with the
00065      *  legacy level of integrity assurance. That is, the probability that the
00066      *  instantaneous URE of the conveying signal exceeds 4.42 times the upper
00067      *  bound value of the current broadcast URA index, for more than 5.2
00068      *  seconds, without an accompanying alert, is less than 1E-5 per hour. If
00069      *  true, indicates that the conveying signal is provided with an enhanced
00070      *  level of integrity assurance. That is, the probability that the
00071      *  instantaneous URE of the conveying signal exceeds 5.73 times the upper
00072      *  bound value of the current broadcast URA index, for more than 5.2
00073      *  seconds, without an accompanying alert, is less than 1E-8 per hour.
00074      */
00075     bool integrity_status_flag{};
00076     bool alert_flag{};         //!< If true, indicates that the SV URA may be worse than indicated in
      d_SV_accuracy, use that SV at our own risk.
00077     bool antispoofing_flag{};  //!< If true, the AntiSpoofing mode is ON in that SV
00078
00079     std::map<int, std::string> satelliteBlock;  //!< Map that stores to which block the PRN belongs
00080
00081     template <class Archive>
00082
00083     /*!
00084      * \brief Serialize is a boost standard method to be called by the boost XML
00085      * serialization. Here is used to save the ephemeris data on disk file.
00086      */
00087     void serialize(Archive& archive, const unsigned int version)
00088     {
00089         using boost::serialization::make_nvp;
00090         if (version)
00091             {
00092             };
00093
00094         archive& BOOST_SERIALIZATION_NVP(PRN);
00095         archive& BOOST_SERIALIZATION_NVP(M_0);
00096         archive& BOOST_SERIALIZATION_NVP(delta_n);
00097         archive& BOOST_SERIALIZATION_NVP(ecc);
00098         archive& BOOST_SERIALIZATION_NVP(sqrtA);
00099         archive& BOOST_SERIALIZATION_NVP(OMEGA_0);
00100         archive& BOOST_SERIALIZATION_NVP(i_0);
00101         archive& BOOST_SERIALIZATION_NVP(omega);
00102         archive& BOOST_SERIALIZATION_NVP(OMEGAdot);
00103         archive& BOOST_SERIALIZATION_NVP(idot);
00104         archive& BOOST_SERIALIZATION_NVP(Cuc);
00105         archive& BOOST_SERIALIZATION_NVP(Cus);
00106         archive& BOOST_SERIALIZATION_NVP(Crc);
00107         archive& BOOST_SERIALIZATION_NVP(Crs);
00108         archive& BOOST_SERIALIZATION_NVP(Cic);
00109         archive& BOOST_SERIALIZATION_NVP(Cis);
00110         archive& BOOST_SERIALIZATION_NVP(toe);
00111         archive& BOOST_SERIALIZATION_NVP(toc);
00112         archive& BOOST_SERIALIZATION_NVP(af0);
00113         archive& BOOST_SERIALIZATION_NVP(af1);
00114         archive& BOOST_SERIALIZATION_NVP(af2);
00115         archive& BOOST_SERIALIZATION_NVP(WN);
00116         archive& BOOST_SERIALIZATION_NVP(tow);
00117         archive& BOOST_SERIALIZATION_NVP(satClkDrift);
00118         archive& BOOST_SERIALIZATION_NVP(dtr);
00119
00120         archive& BOOST_SERIALIZATION_NVP(AODE);
00121         archive& BOOST_SERIALIZATION_NVP(SV_accuracy);
00122         archive& BOOST_SERIALIZATION_NVP(SV_health);
00123         archive& BOOST_SERIALIZATION_NVP(AODC);
00124         archive& BOOST_SERIALIZATION_NVP(TGD1);
00125         archive& BOOST_SERIALIZATION_NVP(TGD2);
00126         archive& BOOST_SERIALIZATION_NVP(sig_type);
```

```
00127         archive& BOOST_SERIALIZATION_NVP(nav_type);
00128         archive& BOOST_SERIALIZATION_NVP(AODO);
00129         archive& BOOST_SERIALIZATION_NVP(fit_interval_flag);
00130         archive& BOOST_SERIALIZATION_NVP(spare1);
00131         archive& BOOST_SERIALIZATION_NVP(spare2);
00132         archive& BOOST_SERIALIZATION_NVP(integrity_status_flag);
00133         archive& BOOST_SERIALIZATION_NVP(alert_flag);
00134         archive& BOOST_SERIALIZATION_NVP(antispoofing_flag);
00135     }
00136 };
00137
00138
00139 /** \} */
00140 /** \} */
00141 #endif  // GNSS_SDR_BEIDOU_DNAV_EPHEMERIS_H
```

13.676 beidou_dnav_iono.h File Reference

Interface of a BEIDOU IONOSPHERIC MODEL storage.

```
#include "gps_iono.h"
#include <boost/serialization/base_object.hpp>
#include <boost/serialization/map.hpp>
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <map>
```

Classes

- struct Beidou_Dnav_Iono_Grid_Point
- class Beidou_Dnav_Iono

  This class is a storage for the BEIDOU IONOSPHERIC data as described in ICD v2.1.

13.676.1 Detailed Description

Interface of a BEIDOU IONOSPHERIC MODEL storage.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_dnav_iono.h.

13.677 beidou_dnav_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_iono.h
00003  * \brief  Interface of a BEIDOU IONOSPHERIC MODEL storage
00004  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_BEIDOU_DNAV_IONO_H
00019 #define GNSS_SDR_BEIDOU_DNAV_IONO_H
00020
00021 #include "gps_iono.h"
00022 #include <boost/serialization/base_object.hpp>
```

```
00023 #include <boost/serialization/map.hpp>
00024 #include <boost/serialization/nvp.hpp>
00025 #include <boost/serialization/version.hpp>
00026 #include <map>
00027
00028 struct Beidou_Dnav_Iono_Grid_Point
00029 {
00030     double vertical_delay{};  //!< B1I vertical delay [m]
00031     int GIVEI{};              //!< Grid Ionospheric Vertical Error Index
00032     bool monitored{};         //!< False when the broadcast delay code is 510
00033     bool available{};         //!< False when the broadcast delay code is 511
00034
00035     template <class Archive>
00036     void serialize(Archive& archive, const unsigned int version)
00037     {
00038         if (version)
00039             {
00040             };
00041         archive& BOOST_SERIALIZATION_NVP(vertical_delay);
00042         archive& BOOST_SERIALIZATION_NVP(GIVEI);
00043         archive& BOOST_SERIALIZATION_NVP(monitored);
00044         archive& BOOST_SERIALIZATION_NVP(available);
00045     }
00046 };
00047
00048 /** \addtogroup Core
00049  * \{ */
00050 /** \addtogroup System_Parameters
00051  * \{ */
00052
00053
00054 /*!
00055  * \brief This class is a storage for the BEIDOU IONOSPHERIC data as described
00056  * in ICD v2.1
00057  */
00058 class Beidou_Dnav_Iono : public Gps_Iono
00059 {
00060 public:
00061     Beidou_Dnav_Iono() = default;  //!< Default constructor
00062
00063     std::map<int, Beidou_Dnav_Iono_Grid_Point> grid_points;  //!< D2 grid corrections, keyed by IGP
      number
00064     bool grid_valid{};
00065
00066     template <class Archive>
00067     void serialize(Archive& archive, const unsigned int version)
00068     {
00069         if (version > 0)
00070             {
00071                 archive& BOOST_SERIALIZATION_BASE_OBJECT_NVP(Gps_Iono);
00072                 archive& BOOST_SERIALIZATION_NVP(grid_points);
00073                 archive& BOOST_SERIALIZATION_NVP(grid_valid);
00074             }
00075         else
00076             {
00077                 // Version 0 archives stored the inherited model directly.
00078                 archive& BOOST_SERIALIZATION_NVP(alpha0);
00079                 archive& BOOST_SERIALIZATION_NVP(alpha1);
00080                 archive& BOOST_SERIALIZATION_NVP(alpha2);
00081                 archive& BOOST_SERIALIZATION_NVP(alpha3);
00082                 archive& BOOST_SERIALIZATION_NVP(beta0);
00083                 archive& BOOST_SERIALIZATION_NVP(beta1);
00084                 archive& BOOST_SERIALIZATION_NVP(beta2);
00085                 archive& BOOST_SERIALIZATION_NVP(beta3);
00086             }
00087     }
00088 };
00089
00090 BOOST_CLASS_VERSION(Beidou_Dnav_Iono, 1)
00091
00092
00093 /** \} */
00094 /** \} */
00095 #endif  // GNSS_SDR_BEIDOU_DNAV_IONO_H
```

13.678 beidou_dnav_navigation_message.h File Reference

Interface of a BeiDou DNAV Data message decoder.

```
#include "Beidou_B1I.h"
#include "Beidou_B3I.h"
#include "Beidou_DNAV.h"
#include "beidou_dnav_almanac.h"
```

```
#include "beidou_dnav_differential_corrections.h"
#include "beidou_dnav_ephemeris.h"
#include "beidou_dnav_iono.h"
#include "beidou_dnav_utc_model.h"
#include <array>
#include <bitset>
#include <cstddef>
#include <cstdint>
#include <map>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Beidou_Dnav_Navigation_Message

  This class decodes BeiDou D1 and D2 NAV data messages.

13.678.1 Detailed Description

Interface of a BeiDou DNAV Data message decoder.

Author

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

Damian Miralles, 2018. dmiralles2009@gmail.com

Carles Fernandez-Prades, 2018-2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_dnav_navigation_message.h.

13.679 beidou_dnav_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_navigation_message.h
00003  * \brief  Interface of a BeiDou DNAV Data message decoder
00004  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00005  * \author Damian Miralles, 2018. dmiralles2009@gmail.com
00006  * \author Carles Fernandez-Prades, 2018-2026. cfernandez(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_BEIDOU_DNAV_NAVIGATION_MESSAGE_H
00021 #define GNSS_SDR_BEIDOU_DNAV_NAVIGATION_MESSAGE_H
00022
00023
00024 #include "Beidou_B1I.h"
00025 #include "Beidou_B3I.h"
00026 #include "Beidou_DNAV.h"
00027 #include "beidou_dnav_almanac.h"
00028 #include "beidou_dnav_differential_corrections.h"
00029 #include "beidou_dnav_ephemeris.h"
00030 #include "beidou_dnav_iono.h"
00031 #include "beidou_dnav_utc_model.h"
00032 #include <array>
00033 #include <bitset>
```

```
00034 #include <cstddef>
00035 #include <cstdint>
00036 #include <map>
00037 #include <string>
00038 #include <utility>
00039 #include <vector>
00040
00041 /** \addtogroup Core
00042  * \{ */
00043 /** \addtogroup System_Parameters
00044  * \{ */
00045
00046
00047 /*!
00048  * \brief This class decodes BeiDou D1 and D2 NAV data messages
00049  */
00050 class Beidou_Dnav_Navigation_Message
00051 {
00052 public:
00053     /*!
00054      * Default constructor
00055      */
00056     Beidou_Dnav_Navigation_Message();
00057
00058     /*!
00059      * \brief Obtain a BDS SV Ephemeris class filled with current SV data
00060      */
00061     Beidou_Dnav_Ephemeris get_ephemeris() const;
00062
00063     /*!
00064      * \brief Obtain a BDS ionospheric correction parameters class filled with current SV data
00065      */
00066     Beidou_Dnav_Iono get_iono();
00067
00068     /*!
00069      * \brief Obtain a BDS UTC model parameters class filled with current SV data
00070      */
00071     Beidou_Dnav_Utc_Model get_utc_model();
00072
00073     /*!
00074      * \brief Obtain the last decoded BeiDou almanac record
00075      */
00076     Beidou_Dnav_Almanac get_almanac() const;
00077
00078     /*!
00079      * \brief Obtain the latest D2 integrity and differential corrections
00080      */
00081     Beidou_Dnav_Differential_Corrections get_differential_corrections() const;
00082
00083     /*!
00084      * \brief Decodes the BDS D1 NAV message
00085      */
00086     int32_t d1_subframe_decoder(std::string const& subframe);
00087
00088     /*!
00089      * \brief Decodes the BDS D2 NAV message
00090      */
00091     int32_t d2_subframe_decoder(std::string const& subframe);
00092
00093     /*!
00094      * \brief Computes the Coordinated Universal Time (UTC) and
00095      * returns it in [s]
00096      */
00097     double utc_time(double beidoutime_corrected) const;
00098
00099     bool satellite_validation();
00100
00101     /*!
00102      * \brief Returns true if new Ephemeris has arrived. The flag is set to false when the function is
      executed
00103      */
00104     bool have_new_ephemeris();
00105
00106     /*!
00107      * \brief Returns true if new Iono model has arrived. The flag is set to false when the function
      is executed
00108      */
00109     bool have_new_iono() const;
00110
00111     /*!
00112      * \brief Returns true if new UTC model has arrived. The flag is set to false when the function is
      executed
00113      */
00114     bool have_new_utc_model();
00115
00116     /*!
00117      * \brief Returns true if new UTC model has arrived. The flag is set to false when the function is
```

```
executed
00118      */
00119     bool have_new_almanac();
00120
00121     /*!
00122      * \brief Returns true when a complete D2 subframes 2-4 correction page has arrived
00123      */
00124     bool have_new_differential_corrections();
00125
00126     /*!
00127      * \brief Sets satellite PRN number
00128      */
00129     inline void set_satellite_PRN(uint32_t prn)
00130     {
00131         i_satellite_PRN = prn;
00132     }
00133
00134     inline void set_signal_type(int32_t signal_type)
00135     {
00136         i_signal_type = signal_type;
00137     }
00138
00139     inline bool get_flag_CRC_test() const
00140     {
00141         return flag_crc_test;
00142     }
00143
00144     inline bool get_flag_new_SOW_available() const
00145     {
00146         return flag_new_SOW_available;
00147     }
00148
00149     inline void set_flag_new_SOW_available(bool new_SOW_available)
00150     {
00151         flag_new_SOW_available = new_SOW_available;
00152     }
00153
00154     inline double get_SOW() const
00155     {
00156         return d_SOW;
00157     }
00158
00159 private:
00160     uint64_t read_navigation_unsigned(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00161     int64_t read_navigation_signed(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00162     uint64_t read_navigation_data_unsigned(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits,
      int32_t first_bit, int32_t logical_offset, int32_t length) const;
00163     int64_t sign_extend(uint64_t value, int32_t length) const;
00164     bool read_navigation_bool(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00165     void print_beidou_word_bytes(uint32_t BEIDOU_word) const;
00166     double wrap_dnav_sow(double sow) const;
00167     bool d1_ephemeris_sow_is_consistent() const;
00168     bool format_check(std::string const& subframe) const;
00169     void clear_d2_ephemeris_page_flags();
00170     bool d2_ephemeris_page_is_expected(int32_t page_ID, double sow);
00171     void advance_d2_ephemeris_page(int32_t page_ID, double sow);
00172     void decode_almanac(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, int32_t prn, int32_t
      amid, bool expanded);
00173     void decode_almanac_health(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, int32_t count,
      int32_t first_prn);
00174     void decode_time_offsets(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits);
00175     void decode_utc_parameters(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits);
00176     void decode_d2_iono_grid(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, int32_t
      page_ID);
00177     void update_d2_bdid(uint64_t value, int32_t first_prn, int32_t count);
00178     void update_d2_udrei(const std::bitset<BEIDOU_DNAV_SUBFRAME_DATA_BITS>& bits, int32_t first_bit,
      int32_t first_slot, int32_t count);
00179
00180     // broadcast orbit 1
00181     double d_SOW{};      // Time of BeiDou Week of the ephemeris set (taken from subframes SOW) [s]
00182     double d_SOW_SF1{};  // Time of BeiDou Week from HOW word of Subframe 1 [s]
00183     double d_SOW_SF2{};  // Time of BeiDou Week from HOW word of Subframe 2 [s]
00184     double d_SOW_SF3{};  // Time of BeiDou Week from HOW word of Subframe 3 [s]
00185     double d_SOW_SF4{};  // Time of BeiDou Week from HOW word of Subframe 4 [s]
00186     double d_SOW_SF5{};  // Time of BeiDou Week from HOW word of Subframe 5 [s]
00187
00188     double d_AODE{};
00189     double d_Crs{};      // Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m]
00190     double d_Delta_n{};  // Mean Motion Difference From Computed Value [semi-circles/s]
00191     double d_M_0{};      // Mean Anomaly at Reference Time [semi-circles]
00192
00193     // broadcast orbit 2
00194     double d_Cuc{};           // Amplitude of the Cosine Harmonic Correction Term to the Argument of
      Latitude [rad]
```

```
00195     double d_eccentricity{};  // Eccentricity [dimensionless]
00196     double d_Cus{};           // Amplitude of the Sine Harmonic Correction Term to the Argument of
      Latitude [rad]
00197     double d_sqrt_A{};        // Square Root of the Semi-Major Axis [sqrt(m)]
00198
00199     // broadcast orbit 3
00200     double d_Toe_sf2{};  // Ephemeris data reference time of week in subframe 2, D1 Message
00201     double d_Toe_sf3{};  // Ephemeris data reference time of week in subframe 3, D1 Message
00202     double d_Toe{};      // Ephemeris data reference time of week in subframe 1, D2 Message
00203     double d_Toc{};      // clock data reference time [s]
00204     double d_Cic{};      // Amplitude of the Cosine Harmonic Correction Term to the Angle of
      Inclination [rad]
00205     double d_OMEGA0{};   // Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
00206     double d_Cis{};      // Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination
      [rad]
00207
00208     // broadcast orbit 4
00209     double d_i_0{};        // Inclination Angle at Reference Time [semi-circles]
00210     double d_Crc{};        // Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m]
00211     double d_OMEGA{};      // Argument of Perigee [semi-cicles]
00212     double d_OMEGA_DOT{};  // Rate of Right Ascension [semi-circles/s]
00213
00214     // broadcast orbit 5
00215     double d_IDOT{};          // Rate of Inclination Angle [semi-circles/s]
00216     int32_t i_BEIDOU_week{};  // BeiDou week number, aka WN [week]
00217
00218     // broadcast orbit 6
00219     int32_t i_SV_accuracy{};  // User Range Accuracy (URA) index of the SV
00220     int32_t i_SV_health{};
00221     double d_TGD1{};  // Estimated Group Delay Differential in B1 [s]
00222     double d_TGD2{};  // Estimated Group Delay Differential in B2 [s]
00223     double d_AODC{};  // Age of Data, Clock
00224
00225     // broadcast orbit 7
00226     // int32_t i_AODO{};              // Age of Data Offset (AODO) term for the navigation message
      correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s]
00227
00228     // bool b_fit_interval_flag{};  // indicates the curve-fit interval used by the CS (Block
      II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4
      hours, 1 = greater than 4 hours.
00229     // double d_spare1{};
00230     // double d_spare2{};
00231
00232     double d_A_f0{};  // Clock correction parameters. Coefficient 0 of code phase offset model [s]
00233     double d_A_f1{};  // Clock correction parameters. Coefficient 1 of code phase offset model [s/s]
00234     double d_A_f2{};  // Clock correction parameters. Coefficient 2 of code phase offset model [s/s^2]
00235
00236     // D2 NAV Message Decoding
00237     uint64_t d_A_f1_msb_bits{};          // Clock correction parameters, D2 NAV MSB
00238     uint64_t d_A_f1_lsb_bits{};          // Clock correction parameters, D2 NAV LSB
00239     uint64_t d_Cuc_msb_bits{};           // Amplitude of the Cosine Harmonic Correction Term to the
      Argument of Latitude [rad]
00240     uint64_t d_Cuc_lsb_bits{};           // Amplitude of the Cosine Harmonic Correction Term to the
      Argument of Latitude [rad]
00241     uint64_t d_eccentricity_msb{};       // Eccentricity [dimensionless]
00242     uint64_t d_eccentricity_lsb{};       // Eccentricity [dimensionless]
00243     uint64_t d_Cic_msb_bits{};           // Amplitude of the Cosine Harmonic Correction Term to the
      Argument of Latitude [rad]
00244     uint64_t d_Cic_lsb_bits{};           // Amplitude of the Cosine Harmonic Correction Term to the
      Argument of Latitude [rad]
00245     uint64_t d_eccentricity_msb_bits{};  // Eccentricity [dimensionless]
00246     uint64_t d_eccentricity_lsb_bits{};
00247     uint64_t d_i_0_msb_bits{};        // Inclination Angle at Reference Time [semi-circles]
00248     uint64_t d_i_0_lsb_bits{};        // Inclination Angle at Reference Time [semi-circles]
00249     uint64_t d_OMEGA_msb_bits{};      // Argument of Perigee [semi-cicles]
00250     uint64_t d_OMEGA_lsb_bits{};      // Argument of Perigee [semi-cicles]
00251     uint64_t d_OMEGA_DOT_msb_bits{};  // Rate of Right Ascension [semi-circles/s]
00252     uint64_t d_OMEGA_DOT_lsb_bits{};  // Rate of Right Ascension [semi-circles/s]
00253
00254     // D2 integrity and differential corrections
00255     std::array<bool, 63> d_d2_bdid{};
00256     std::array<int32_t, 24> d_d2_rurai{};
00257     std::array<int32_t, 24> d_d2_udrei{};
00258     std::array<double, 24> d_d2_delta_t{};
00259     std::array<bool, 24> d_d2_delta_t_available{};
00260     std::array<bool, 24> d_d2_correction_valid{};
00261     int32_t i_SatH2{};
00262     int32_t i_BDEpID{};
00263     int32_t i_d2_integrity_page{};
00264     double d_d2_integrity_sow{};
00265     bool flag_d2_sf2_valid{};
00266     bool flag_d2_sf3_valid{};
00267     bool flag_new_differential_corrections{};
00268
00269     // Almanac
00270     // double d_Toa{};                          // Almanac reference time [s]
00271     // int32_t i_WN_A{};                        // Modulo 256 of the GPS week number to which the
```

```
      almanac reference time (d_Toa) is referenced
00272     std::map<int32_t, int32_t> almanacHealth;  // Map that stores the health information stored in the
      almanac
00273     Beidou_Dnav_Almanac d_almanac;
00274     int32_t i_AmEpID{};
00275     bool flag_new_almanac{};
00276     bool flag_almanac_week_valid{};
00277
00278     std::map<int32_t, std::string> satelliteBlock;  // Map that stores to which block the PRN belongs
00279
00280     // satellite identification info
00281     int32_t i_signal_type{};  // BDS: data source (0:unknown,1:B1I,2:B1Q,3:B2I,4:B2Q,5:B3I,6:B3Q)
00282     uint32_t i_satellite_PRN{};
00283
00284     // Ionospheric parameters
00285     double d_alpha0{};  // Coefficient 0 of a cubic equation representing the amplitude of the
      vertical delay [s]
00286     double d_alpha1{};  // Coefficient 1 of a cubic equation representing the amplitude of the
      vertical delay [s/semi-circle]
00287     double d_alpha2{};  // Coefficient 2 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^2]
00288     double d_alpha3{};  // Coefficient 3 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^3]
00289     double d_beta0{};   // Coefficient 0 of a cubic equation representing the period of the model [s]
00290     double d_beta1{};   // Coefficient 1 of a cubic equation representing the period of the model
      [s/semi-circle]
00291     double d_beta2{};   // Coefficient 2 of a cubic equation representing the period of the model
      [s(semi-circle)^2]
00292     double d_beta3{};   // Coefficient 3 of a cubic equation representing the period of the model
      [s(semi-circle)^3]
00293     std::map<int, Beidou_Dnav_Iono_Grid_Point> d_iono_grid;
00294     bool flag_iono_grid_valid{};
00295
00296     // UTC parameters
00297     double d_A1UTC{};       // 1st order term of a model that relates GPS and UTC time [s/s]
00298     double d_A0UTC{};       // Constant of a model that relates GPS and UTC time [s]
00299     int32_t i_DeltaT_LS{};  // BDT-UTC delta time due to leap seconds [s]
00300     int32_t i_WN_LSF{};     // Week number at the end of which the leap second becomes effective
      [weeks]
00301     int32_t i_DN{};         // Day number (DN) at the end of which the leap second becomes effective
      [days]
00302     double d_DeltaT_LSF{};  // Scheduled future or recent past (relative to NAV message upload) value
      of the delta time due to leap seconds [s]
00303     double d_A1GPS{};
00304     double d_A0GPS{};
00305     double d_A1GAL{};
00306     double d_A0GAL{};
00307     double d_A1GLO{};
00308     double d_A0GLO{};
00309
00310     double d_SQRT_A_ALMANAC{};
00311     double d_A1_ALMANAC{};
00312     double d_A0_ALMANAC{};
00313     double d_OMEGA0_ALMANAC{};
00314     double d_E_ALMANAC{};
00315     double d_DELTA_I{};
00316     double d_TOA{};
00317     double d_OMEGA_DOT_ALMANAC{};
00318     double d_OMEGA_ALMANAC{};
00319     double d_M0_ALMANAC{};
00320     int32_t almanac_WN{};
00321     double d_toa2{};
00322
00323     // System flags for data processing
00324     bool flag_eph_valid{};
00325     bool flag_utc_model_valid{};
00326     bool flag_iono_valid{};
00327     bool flag_d1_sf1{};
00328     bool flag_d1_sf2{};
00329     bool flag_d1_sf3{};
00330     bool flag_d1_sf4{};
00331     bool flag_d1_sf5{};
00332     bool flag_new_SOW_available{};
00333     bool flag_crc_test{};
00334     double d_previous_aode{-1.0};
00335
00336     // bool flag_d1_sf5_p7{};   // D1 NAV Message, Subframe 5, Page 09 decoded indicator
00337     // bool flag_d1_sf5_p8{};   // D1 NAV Message, Subframe 5, Page 09 decoded indicator
00338     bool flag_d1_sf5_p9{};   // D1 NAV Message, Subframe 5, Page 09 decoded indicator
00339     bool flag_d1_sf5_p10{};  // D1 NAV Message, Subframe 5, Page 10 decoded indicator
00340
00341     bool flag_sf1_p1{};   // D2 NAV Message, Subframe 1, Page 1 decoded indicator
00342     bool flag_sf1_p2{};   // D2 NAV Message, Subframe 1, Page 2 decoded indicator
00343     bool flag_sf1_p3{};   // D2 NAV Message, Subframe 1, Page 3 decoded indicator
00344     bool flag_sf1_p4{};   // D2 NAV Message, Subframe 1, Page 4 decoded indicator
00345     bool flag_sf1_p5{};   // D2 NAV Message, Subframe 1, Page 5 decoded indicator
00346     bool flag_sf1_p6{};   // D2 NAV Message, Subframe 1, Page 6 decoded indicator
```

```
00347     bool flag_sf1_p7{};   // D2 NAV Message, Subframe 1, Page 7 decoded indicator
00348     bool flag_sf1_p8{};   // D2 NAV Message, Subframe 1, Page 8 decoded indicator
00349     bool flag_sf1_p9{};   // D2 NAV Message, Subframe 1, Page 9 decoded indicator
00350     bool flag_sf1_p10{};  // D2 NAV Message, Subframe 1, Page 10 decoded indicator
00351     bool flag_d2_ephemeris_collection_started{};
00352     int32_t d_d2_expected_page{1};
00353     double d_d2_expected_sow{};
00354 };
00355
00356
00357 /** \} */
00358 /** \} */
00359 #endif  // GNSS_SDR_BEIDOU_DNAV_NAVIGATION_MESSAGE_H
```

13.680 beidou_dnav_utc_model.h File Reference

Interface of a BeiDou UTC MODEL storage.

`#include <boost/serialization/nvp.hpp>`

Classes

- class Beidou_Dnav_Utc_Model

  This class is a storage for the BeiDou DNAV UTC Model.

13.680.1 Detailed Description

Interface of a BeiDou UTC MODEL storage.

Author

Damian Miralles, 2018. dmiralles2009@gmail.com

Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file beidou_dnav_utc_model.h.

13.681 beidou_dnav_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_dnav_utc_model.h
00003  * \brief  Interface of a BeiDou UTC MODEL storage
00004  * \author Damian Miralles, 2018. dmiralles2009@gmail.com
00005  * \author Sergi Segura, 2018. sergi.segura.munoz(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_BEIDOU_DNAV_UTC_MODEL_H
00020 #define GNSS_SDR_BEIDOU_DNAV_UTC_MODEL_H
00021
00022 #include <boost/serialization/nvp.hpp>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class is a storage for the BeiDou DNAV UTC Model.
```

```
00032  * \details Implementation follows the interface described in the Open Service Signal (Version 2.1)
00033  *
00034  */
00035 class Beidou_Dnav_Utc_Model
00036 {
00037 public:
00038     Beidou_Dnav_Utc_Model() = default;
00039
00040     // BeiDou UTC parameters
00041     double A0_UTC{};      //!< BDT clock bias relative to UTC [s]
00042     double A1_UTC{};      //!< BDT clock rate relative to UTC [s/s]
00043     int DeltaT_LS{};      //!< Delta time due to leap seconds before the new leap second effective
00044     int WN_LSF{};         //!< Week number of the new leap second
00045     int DN{};             //!< Day number of week of the new leap second
00046     double DeltaT_LSF{};  //!< Delta time due to leap seconds after the new leap second effective [s]
00047
00048     // BeiDou to GPS time corrections
00049     double A0_GPS{};  //!< BDT clock bias relative to GPS time [s]
00050     double A1_GPS{};  //!< BDT clock rate relative to GPS time [s/s]
00051
00052     // BeiDou to Galileo time corrections
00053     double A0_GAL{};  //!< BDT clock bias relative to GAL time [s]
00054     double A1_GAL{};  //!< BDT clock rate relative to GAL time [s/s]
00055
00056     // BeiDou to GLONASS time corrections
00057     double A0_GLO{};  //!< BDT clock bias relative to GLO time [s]
00058     double A1_GLO{};  //!< BDT clock rate relative to GLO time [s/s]
00059
00060     bool valid{};
00061
00062     template <class Archive>
00063     /*
00064      * \brief Serialize is a boost standard method to be called by the boost XML
00065      * serialization. Here is used to save the ephemeris data on disk file.
00066      */
00067     inline void serialize(Archive& archive, const unsigned int version)
00068     {
00069         if (version)
00070             {
00071             };
00072         archive& BOOST_SERIALIZATION_NVP(A1_UTC);
00073         archive& BOOST_SERIALIZATION_NVP(A0_UTC);
00074         archive& BOOST_SERIALIZATION_NVP(DeltaT_LS);
00075         archive& BOOST_SERIALIZATION_NVP(WN_LSF);
00076         archive& BOOST_SERIALIZATION_NVP(DN);
00077         archive& BOOST_SERIALIZATION_NVP(DeltaT_LSF);
00078         archive& BOOST_SERIALIZATION_NVP(A0_GPS);
00079         archive& BOOST_SERIALIZATION_NVP(A1_GPS);
00080         archive& BOOST_SERIALIZATION_NVP(A0_GAL);
00081         archive& BOOST_SERIALIZATION_NVP(A1_GAL);
00082         archive& BOOST_SERIALIZATION_NVP(A0_GLO);
00083         archive& BOOST_SERIALIZATION_NVP(A1_GLO);
00084         archive& BOOST_SERIALIZATION_NVP(valid);
00085     }
00086 };
00087
00088
00089 /** \} */
00090 /** \} */
00091 #endif  // GNSS_SDR_BEIDOU_DNAV_UTC_MODEL_H
```

13.682 beidou_ldpc.h File Reference

Shared BeiDou B-CNAV1/B-CNAV2 non-binary LDPC decoder over GF(64).

```
#include <cstddef>
#include <cstdint>
#include <vector>
```

Classes

- struct BeidouLdpcGraph

Functions

- bool GaloisField64::valid_symbol (uint8_t x)
- uint8_t GaloisField64::add (uint8_t a, uint8_t b)

- uint8_t GaloisField64::mul (uint8_t a, uint8_t b)
- uint8_t GaloisField64::inv (uint8_t a)
- bool **beidou_ldpc_init_graph** (int32_t num_checks, int32_t num_variables, int32_t row_weight, const uint16_t *icd_index, const uint8_t *icd_element, int32_t num_entries, BeidouLdpcGraph &graph)
- bool **beidou_ldpc_decode** (const BeidouLdpcGraph &graph, const float *bit_llr, int32_t num_bits, uint8_t *info_bits, uint8_t *codeword_bits, bool enable_sum_product=false)

Variables

- constexpr int32_t BEIDOU_LDPC_NM = 8
- constexpr int32_t BEIDOU_LDPC_MAX_ITER = 15
- constexpr uint8_t BEIDOU_GF64_EXP [64]
- constexpr int8_t BEIDOU_GF64_LOG [64]
- constexpr uint8_t GaloisField64::kZero = 0U
- constexpr uint8_t GaloisField64::kOrder = 63U
- constexpr uint8_t GaloisField64::kFieldSize = 64U

13.682.1 Detailed Description

Shared BeiDou B-CNAV1/B-CNAV2 non-binary LDPC decoder over GF(64).

Author

Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file beidou_ldpc.h.

13.682.2 Function Documentation

13.682.2.1 add()

```
uint8_t GaloisField64::add (
            uint8_t a,
            uint8_t b)  [inline]
```

Definition at line 52 of file beidou_ldpc.h.

13.682.2.2 inv()

```
uint8_t GaloisField64::inv (
            uint8_t a)  [inline]
```

Definition at line 74 of file beidou_ldpc.h.

13.682.2.3 mul()

```
uint8_t GaloisField64::mul (
            uint8_t a,
            uint8_t b)  [inline]
```

Definition at line 62 of file beidou_ldpc.h.

13.682.2.4 valid_symbol()

```
bool GaloisField64::valid_symbol (
            uint8_t x)  [inline]
```

Definition at line 46 of file beidou_ldpc.h.

13.682.3 Variable Documentation

13.682.3.1 BEIDOU_GF64_EXP

```
uint8_t BEIDOU_GF64_EXP[64]  [constexpr]
```

Initial value:

```
                                = {
1, 2, 4, 8, 16, 32, 3, 6, 12, 24, 48, 35, 5, 10, 20, 40, 19, 38, 15, 30, 60, 59, 53,
41, 17, 34, 7, 14, 28, 56, 51, 37, 9, 18, 36, 11, 22, 44, 27, 54, 47, 29, 58, 55,
45, 25, 50, 39, 13, 26, 52, 43, 21, 42, 23, 46, 31, 62, 63, 61, 57, 49, 33, 1}
```

Definition at line 28 of file beidou_ldpc.h.

13.682.3.2 BEIDOU_GF64_LOG

```
int8_t BEIDOU_GF64_LOG[64]  [constexpr]
```

Initial value:

```
                                = {
0, 0, 1, 6, 2, 12, 7, 26, 3, 32, 13, 35, 8, 48, 27, 18, 4, 24, 33, 16, 14, 52, 36,
54, 9, 45, 49, 38, 28, 41, 19, 56, 5, 62, 25, 11, 34, 31, 17, 47, 15, 23, 53, 51,
37, 44, 55, 40, 10, 61, 46, 30, 50, 22, 39, 43, 29, 60, 42, 21, 20, 59, 57, 58}
```

Definition at line 34 of file beidou_ldpc.h.

13.682.3.3 BEIDOU_LDPC_MAX_ITER

```
int32_t BEIDOU_LDPC_MAX_ITER = 15  [constexpr]
```

Definition at line 25 of file beidou_ldpc.h.

13.682.3.4 BEIDOU_LDPC_NM

```
int32_t BEIDOU_LDPC_NM = 8  [constexpr]
```

Definition at line 24 of file beidou_ldpc.h.

13.682.3.5 kFieldSize

```
uint8_t GaloisField64::kFieldSize = 64U  [constexpr]
```

Definition at line 43 of file beidou_ldpc.h.

13.682.3.6 kOrder

```
uint8_t GaloisField64::kOrder = 63U  [constexpr]
```

Definition at line 42 of file beidou_ldpc.h.

13.682.3.7 kZero

```
uint8_t GaloisField64::kZero = 0U  [constexpr]
```

Definition at line 41 of file beidou_ldpc.h.

13.683 beidou_ldpc.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file beidou_ldpc.h
00003  * \brief Shared BeiDou B-CNAV1/B-CNAV2 non-binary LDPC decoder over GF(64)
00004  * \author Wenhao Ou, 2026. ouwh(at)mail2.sysu.edu.cn
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_BEIDOU_LDPC_H
```

```
00018 #define GNSS_SDR_BEIDOU_LDPC_H
00019
00020 #include <cstddef>
00021 #include <cstdint>
00022 #include <vector>
00023
00024 constexpr int32_t BEIDOU_LDPC_NM = 8;
00025 constexpr int32_t BEIDOU_LDPC_MAX_ITER = 15;
00026
00027 // B1C and B2a use p(x) = x^6 + x + 1, with MSB-first vector representation.
00028 constexpr uint8_t BEIDOU_GF64_EXP[64] = {
00029     1, 2, 4, 8, 16, 32, 3, 6, 12, 24, 48, 35, 5, 10, 20, 40, 19, 38, 15, 30, 60, 59, 53,
00030     41, 17, 34, 7, 14, 28, 56, 51, 37, 9, 18, 36, 11, 22, 44, 27, 54, 47, 29, 58, 55,
00031     45, 25, 50, 39, 13, 26, 52, 43, 21, 42, 23, 46, 31, 62, 63, 61, 57, 49, 33, 1};
00032
00033
00034 constexpr int8_t BEIDOU_GF64_LOG[64] = {
00035     0, 0, 1, 6, 2, 12, 7, 26, 3, 32, 13, 35, 8, 48, 27, 18, 4, 24, 33, 16, 14, 52, 36,
00036     54, 9, 45, 49, 38, 28, 41, 19, 56, 5, 62, 25, 11, 34, 31, 17, 47, 15, 23, 53, 51,
00037     37, 44, 55, 40, 10, 61, 46, 30, 50, 22, 39, 43, 29, 60, 42, 21, 20, 59, 57, 58};
00038
00039 namespace GaloisField64
00040 {
00041 constexpr uint8_t kZero = 0U;
00042 constexpr uint8_t kOrder = 63U;
00043 constexpr uint8_t kFieldSize = 64U;
00044
00045 // LUT values follow the B1C and B2a ICD Annex GF(2^6) mapping rules.
00046 inline bool valid_symbol(uint8_t x)
00047 {
00048     return x < kFieldSize;
00049 }
00050
00051
00052 inline uint8_t add(uint8_t a, uint8_t b)
00053 {
00054     if (!valid_symbol(a) || !valid_symbol(b))
00055         {
00056             return kZero;
00057         }
00058     return static_cast<uint8_t>(a ^ b);
00059 }
00060
00061
00062 inline uint8_t mul(uint8_t a, uint8_t b)
00063 {
00064     if (!valid_symbol(a) || !valid_symbol(b) || a == kZero || b == kZero)
00065         {
00066             return kZero;
00067         }
00068     const int32_t log_sum = static_cast<int32_t>(BEIDOU_GF64_LOG[a]) +
00069                             static_cast<int32_t>(BEIDOU_GF64_LOG[b]);
00070     return BEIDOU_GF64_EXP[log_sum % kOrder];
00071 }
00072
00073
00074 inline uint8_t inv(uint8_t a)
00075 {
00076     if (!valid_symbol(a) || a == kZero)
00077         {
00078             return kZero;
00079         }
00080     const int32_t exponent = static_cast<int32_t>(kOrder) - static_cast<int32_t>(BEIDOU_GF64_LOG[a]);
00081     return BEIDOU_GF64_EXP[exponent % kOrder];
00082 }
00083 }  // namespace GaloisField64
00084
00085
00086 struct BeidouLdpcGraph
00087 {
00088     int32_t num_checks = 0;
00089     int32_t num_variables = 0;
00090     int32_t row_weight = 0;
00091
00092     // Check-node (CSR-like) adjacency: [check_offsets[i], check_offsets[i+1]).
00093     std::vector<uint32_t> check_offsets;
00094     std::vector<uint16_t> check_to_var;
00095     std::vector<uint8_t> check_to_h;
00096
00097     // Variable-node (CSC-like) adjacency: [var_offsets[j], var_offsets[j+1]).
00098     std::vector<uint32_t> var_offsets;
00099     std::vector<uint16_t> var_to_check;
00100     std::vector<uint32_t> var_to_edge;
00101     std::vector<uint8_t> var_to_h;
00102     std::vector<uint8_t> var_to_h_inv;
00103 };
00104
```

```
00105
00106 // ICD tables are laid out in four bands of four entries per printed row.
00107 // Only row weight four is supported (ICD table layout).
00108 bool beidou_ldpc_init_graph(
00109     int32_t num_checks,
00110     int32_t num_variables,
00111     int32_t row_weight,
00112     const uint16_t* icd_index,
00113     const uint8_t* icd_element,
00114     int32_t num_entries,
00115     BeidouLdpcGraph& graph);
00116
00117
00118 // Positive bit LLRs favor 1. Bits within each GF(64) symbol are MSB first.
00119 // Supply at least one output: n-m information symbols or all n codeword symbols.
00120 // Outputs are written only after the parity checks pass; CRC validation is separate.
00121 // Bit LLRs should be calibrated (2|y|/sigma^2 for BPSK, as in the ICD Annex): the
00122 // EMS truncation offset is tuned for that scale and sum-product assumes it.
00123 // Decoding uses the Extended Min-Sum algorithm of the ICD Annex. The optional
00124 // full-alphabet sum-product retry adds up to BEIDOU_LDPC_MAX_ITER iterations
00125 // after an EMS failure.
00126 bool beidou_ldpc_decode(const BeidouLdpcGraph& graph, const float* bit_llr,
00127     int32_t num_bits, uint8_t* info_bits, uint8_t* codeword_bits, bool enable_sum_product = false);
00128
00129 #endif  // GNSS_SDR_BEIDOU_LDPC_H
```

13.684 display.h File Reference

Defines useful display constants.
`#include <string>`

Macros

- #define DISPLAY_COLORS 1

Variables

- const std::string TEXT_RESET = "\033[0m"
- const std::string TEXT_BLACK = "\033[30m"
- const std::string TEXT_RED = "\033[31m"
- const std::string TEXT_GREEN = "\033[32m"
- const std::string TEXT_YELLOW = "\033[33m"
- const std::string TEXT_BLUE = "\033[34m"
- const std::string TEXT_MAGENTA = "\033[35m"
- const std::string TEXT_CYAN = "\033[36m"
- const std::string TEXT_WHITE = "\033[37m"
- const std::string TEXT_BOLD_BLACK = "\033[1m\033[30m"
- const std::string TEXT_BOLD_RED = "\033[1m\033[31m"
- const std::string TEXT_BOLD_GREEN = "\033[1m\033[32m"
- const std::string TEXT_BOLD_YELLOW = "\033[1m\033[33m"
- const std::string TEXT_BOLD_BLUE = "\033[1m\033[34m"
- const std::string TEXT_BOLD_MAGENTA = "\033[1m\033[35m"
- const std::string TEXT_BOLD_CYAN = "\033[1m\033[36m"
- const std::string TEXT_BOLD_WHITE = "\033[1m\033[37m"

13.684.1 Detailed Description

Defines useful display constants.

Author

Antonio Ramos, 2018. antonio.ramos(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file display.h.

13.685 display.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file display.h
00003  * \brief  Defines useful display constants
00004  * \author Antonio Ramos, 2018. antonio.ramos(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_DISPLAY_H
00018 #define GNSS_SDR_DISPLAY_H
00019
00020 #include <string>
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027
00028 #ifndef NO_DISPLAY_COLORS
00029 #define DISPLAY_COLORS 1
00030 #endif
00031
00032
00033 #ifdef DISPLAY_COLORS
00034
00035 const std::string TEXT_RESET = "\033[0m";
00036 const std::string TEXT_BLACK = "\033[30m";
00037 const std::string TEXT_RED = "\033[31m";
00038 const std::string TEXT_GREEN = "\033[32m";
00039 const std::string TEXT_YELLOW = "\033[33m";
00040 const std::string TEXT_BLUE = "\033[34m";
00041 const std::string TEXT_MAGENTA = "\033[35m";
00042 const std::string TEXT_CYAN = "\033[36m";
00043 const std::string TEXT_WHITE = "\033[37m";
00044 const std::string TEXT_BOLD_BLACK = "\033[1m\033[30m";
00045 const std::string TEXT_BOLD_RED = "\033[1m\033[31m";
00046 const std::string TEXT_BOLD_GREEN = "\033[1m\033[32m";
00047 const std::string TEXT_BOLD_YELLOW = "\033[1m\033[33m";
00048 const std::string TEXT_BOLD_BLUE = "\033[1m\033[34m";
00049 const std::string TEXT_BOLD_MAGENTA = "\033[1m\033[35m";
00050 const std::string TEXT_BOLD_CYAN = "\033[1m\033[36m";
00051 const std::string TEXT_BOLD_WHITE = "\033[1m\033[37m";
00052
00053 #else
00054
00055 const std::string TEXT_RESET = "";
00056 const std::string TEXT_BLACK = "";
00057 const std::string TEXT_RED = "";
00058 const std::string TEXT_GREEN = "";
00059 const std::string TEXT_YELLOW = "";
00060 const std::string TEXT_BLUE = "";
00061 const std::string TEXT_MAGENTA = "";
00062 const std::string TEXT_CYAN = "";
00063 const std::string TEXT_WHITE = "";
00064 const std::string TEXT_BOLD_BLACK = "";
00065 const std::string TEXT_BOLD_RED = "";
00066 const std::string TEXT_BOLD_GREEN = "";
00067 const std::string TEXT_BOLD_YELLOW = "";
00068 const std::string TEXT_BOLD_BLUE = "";
00069 const std::string TEXT_BOLD_MAGENTA = "";
00070 const std::string TEXT_BOLD_CYAN = "";
00071 const std::string TEXT_BOLD_WHITE = "";
00072
00073 #endif  // DISPLAY_COLORS
00074
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_DISPLAY_H
```

13.686 galileo_almanac.h File Reference

Interface of a Galileo ALMANAC storage.

```
#include "gnss_almanac.h"
#include <boost/serialization/nvp.hpp>
```

Classes

- class Galileo_Almanac

  This class is a storage for the Galileo SV ALMANAC data.

13.686.1 Detailed Description

Interface of a Galileo ALMANAC storage.

Author

Carles Fernandez, 2018. cfernandez(at)cttc.cat

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_almanac.h.

13.687 galileo_almanac.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_almanac.h
00003  * \brief  Interface of a Galileo ALMANAC storage
00004  * \author Carles Fernandez, 2018. cfernandez(at)cttc.cat
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GALILEO_ALMANAC_H
00019 #define GNSS_SDR_GALILEO_ALMANAC_H
00020
00021 #include "gnss_almanac.h"
00022 #include <boost/serialization/nvp.hpp>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class is a storage for the Galileo SV ALMANAC data
00032  */
00033 class Galileo_Almanac : public Gnss_Almanac
00034 {
00035 public:
00036     /*!
00037      * Default constructor
00038      */
00039     Galileo_Almanac()
00040     {
00041         this->System = 'E';
00042     };
00043
00044     int32_t IODa{};
00045     int32_t E5b_HS{};
00046     int32_t E1B_HS{};
00047     int32_t E5a_HS{};
00048
```

```
00049     template <class Archive>
00050
00051     void serialize(Archive& ar, const unsigned int version)
00052     {
00053         if (version)
00054             {
00055             };
00056         ar& BOOST_SERIALIZATION_NVP(PRN);
00057         ar& BOOST_SERIALIZATION_NVP(delta_i);
00058         ar& BOOST_SERIALIZATION_NVP(toa);
00059         ar& BOOST_SERIALIZATION_NVP(WNa);
00060         ar& BOOST_SERIALIZATION_NVP(M_0);
00061         ar& BOOST_SERIALIZATION_NVP(ecc);
00062         ar& BOOST_SERIALIZATION_NVP(sqrtA);
00063         ar& BOOST_SERIALIZATION_NVP(OMEGA_0);
00064         ar& BOOST_SERIALIZATION_NVP(omega);
00065         ar& BOOST_SERIALIZATION_NVP(OMEGAdot);
00066         ar& BOOST_SERIALIZATION_NVP(af0);
00067         ar& BOOST_SERIALIZATION_NVP(af1);
00068         ar& BOOST_SERIALIZATION_NVP(IODa);
00069         ar& BOOST_SERIALIZATION_NVP(E5b_HS);
00070         ar& BOOST_SERIALIZATION_NVP(E1B_HS);
00071         ar& BOOST_SERIALIZATION_NVP(E5a_HS);
00072     }
00073 };
00074
00075
00076 /** \} */
00077 /** \} */
00078 #endif  // GNSS_SDR_GALILEO_ALMANAC_H
```

13.688 galileo_almanac_helper.h File Reference

Interface of a Galileo ALMANAC storage helper.

```
#include "galileo_almanac.h"
#include <cstdint>
```

Classes

- class Galileo_Almanac_Helper

  This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD.

13.688.1 Detailed Description

Interface of a Galileo ALMANAC storage helper.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

13.688.1.1 autotoc_md684

Author

Mara Branzanti 2013. mara.branzanti(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_almanac_helper.h.

13.689 galileo_almanac_helper.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_almanac_helper.h
00003  * \brief  Interface of a Galileo ALMANAC storage helper
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  * -------------------------------------------------------------------------
00007  *
```

```
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GALILEO_ALMANAC_HELPER_H
00018 #define GNSS_SDR_GALILEO_ALMANAC_HELPER_H
00019
00020 #include "galileo_almanac.h"
00021 #include <cstdint>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief This class is a storage for the GALILEO ALMANAC data as described in GALILEO ICD
00031  *
00032  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
     paragraph 5.1.10
00033  */
00034 class Galileo_Almanac_Helper
00035 {
00036 public:
00037     Galileo_Almanac_Helper() = default;  //!< Default constructor
00038
00039     Galileo_Almanac get_almanac(int i) const;
00040
00041     // Word type 7: Almanac for SVID1 (1/2), almanac reference time and almanac reference week number
00042     int32_t IOD_a_7{};
00043     int32_t WN_a_7{};
00044     int32_t t0a_7{};
00045     int32_t SVID1_7{};
00046     double DELTA_A_7{};
00047     double e_7{};
00048     double omega_7{};
00049     double delta_i_7{};
00050     double Omega0_7{};
00051     double Omega_dot_7{};
00052     double M0_7{};
00053
00054     // Word type 8: Almanac for SVID1 (2/2) and SVID2 (1/2)
00055     int32_t IOD_a_8{};
00056     double af0_8{};
00057     double af1_8{};
00058     int32_t E5b_HS_8{};
00059     int32_t E1B_HS_8{};
00060     int32_t E5a_HS_8{};
00061     int32_t SVID2_8{};
00062     double DELTA_A_8{};
00063     double e_8{};
00064     double omega_8{};
00065     double delta_i_8{};
00066     double Omega0_8{};
00067     double Omega_dot_8{};
00068
00069     // Word type 9: Almanac for SVID2 (2/2) and SVID3 (1/2)
00070     int32_t IOD_a_9{};
00071     int32_t WN_a_9{};
00072     int32_t t0a_9{};
00073     double M0_9{};
00074     double af0_9{};
00075     double af1_9{};
00076     int32_t E5b_HS_9{};
00077     int32_t E1B_HS_9{};
00078     int32_t E5a_HS_9{};
00079     int32_t SVID3_9{};
00080     double DELTA_A_9{};
00081     double e_9{};
00082     double omega_9{};
00083     double delta_i_9{};
00084
00085     // Word type 10: Almanac for SVID3 (2/2)
00086     int32_t IOD_a_10{};
00087     double Omega0_10{};
00088     double Omega_dot_10{};
00089     double M0_10{};
00090     double af0_10{};
00091     double af1_10{};
00092     int32_t E5b_HS_10{};
00093     int32_t E1B_HS_10{};
```

```
00094     int32_t E5a_HS_10{};
00095 };
00096
00097
00098 /** \} */
00099 /** \} */
00100 #endif  // GNSS_SDR_GALILEO_ALMANAC_HELPER_H
```

13.690 Galileo_CNAV.h File Reference

Galileo CNAV message constants. Data from: Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

```
#include <cstddef>
#include <cstdint>
#include <limits>
#include <string>
#include <unordered_map>
#include <utility>
```

Functions

- const std::pair< int32_t, int32_t > **GALILEO_HAS_STATUS** ({1, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_RESERVED** ({3, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_TYPE** ({5, 2})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_ID** ({7, 5})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_SIZE** ({12, 5})
- const std::pair< int32_t, int32_t > **GALILEO_HAS_MESSAGE_PAGE_ID** ({17, 8})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_TOH** ({1, 12})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_MASK_FLAG** ({13, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_ORBIT_CORRECTION_FLAG** ({14, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CLOCK_FULLSET_FLAG** ({15, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CLOCK_SUBSET_FLAG** ({16, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_CODE_BIAS_FLAG** ({17, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_PHASE_BIAS_FLAG** ({18, 1})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_RESERVED** ({19, 4})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_MASK_ID** ({23, 5})
- const std::pair< int32_t, int32_t > **GALILEO_MT1_HEADER_IOD_SET_ID** ({28, 5})

Variables

- constexpr size_t HAS_MSG_NSYS_LENGTH = 4
- constexpr size_t HAS_MSG_ID_MASK_LENGTH = 4
- constexpr size_t HAS_MSG_SATELLITE_MASK_LENGTH = 40
- constexpr size_t HAS_MSG_SIGNAL_MASK_LENGTH = 16
- constexpr size_t HAS_MSG_NAV_MESSAGE_LENGTH = 3
- constexpr size_t HAS_MSG_MASK_RESERVED_LENGTH = 6
- constexpr size_t HAS_MSG_VALIDITY_INDEX_LENGTH = 4
- constexpr size_t HAS_MSG_IOD_GPS_LENGTH = 8
- constexpr size_t HAS_MSG_IOD_GAL_LENGTH = 10
- constexpr size_t HAS_MSG_DELTA_RADIAL_LENGTH = 13
- constexpr size_t HAS_MSG_DELTA_IN_TRACK_LENGTH = 12
- constexpr size_t HAS_MSG_DELTA_CROSS_TRACK_LENGTH = 12
- constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH = 2
- constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH = 13
- constexpr size_t HAS_MSG_NSYS_SUB_LENGTH = 4
- constexpr size_t HAS_MSG_ID_CLOCK_SUBSET_LENGTH = 4
- constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH = 2

- constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH = 13
- constexpr size_t HAS_MSG_CODE_BIAS_LENGTH = 11
- constexpr size_t HAS_MSG_PHASE_BIAS_LENGTH = 11
- constexpr size_t HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH = 2
- constexpr uint64_t MAX_SECONDS_REMEMBERING_MID = 150
- constexpr uint64_t MAX_SECONDS_REMEMBERING_MASK_IOD = 1800
- constexpr uint32_t GALILEO_HAS_SECONDS_PER_HOUR = 3600U
- constexpr uint32_t GALILEO_HAS_SECONDS_PER_WEEK = 604800U
- constexpr uint32_t GALILEO_HAS_HALF_WEEK_SECONDS = GALILEO_HAS_SECONDS_PER_WEEK / 2U
- constexpr uint32_t GALILEO_HAS_INVALID_WEEK = std::numeric_limits<uint32_t>::max()
- constexpr uint32_t GALILEO_HAS_INVALID_TOW = std::numeric_limits<uint32_t>::max()
- constexpr int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000
- constexpr int32_t GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS = 1000
- constexpr int32_t GALILEO_CNAV_PAGE_MS = 1
- constexpr int32_t GALILEO_CNAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_CNAV_INTERLEAVER_COLS = 123
- constexpr int32_t GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND = 1000
- constexpr int32_t GALILEO_CNAV_HAS_PAGE_DATA_BITS = 448
- constexpr int32_t GALILEO_CNAV_PAGE_RESERVED_BITS = 14
- constexpr int32_t GALILEO_CNAV_BYTES_FOR_CRC = 58
- constexpr int32_t GALILEO_CNAV_CRC_LENGTH = 24
- constexpr int32_t GALILEO_CNAV_MESSAGE_BITS_PER_PAGE = 424
- constexpr int32_t GALILEO_CNAV_PAGE_HEADER_BITS = 24
- constexpr int32_t GALILEO_CNAV_PREAMBLE_LENGTH_BITS = 16
- constexpr int32_t GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK = 255
- constexpr int32_t GALILEO_CNAV_MT1_HEADER_BITS = 32
- constexpr int32_t GALILEO_CNAV_OCTETS_IN_SUBPAGE = 53
- constexpr int32_t GALILEO_CNAV_INFORMATION_VECTOR_LENGTH = 32
- constexpr int32_t GALILEO_CNAV_BITS_FOR_CRC = GALILEO_CNAV_HAS_PAGE_DATA_BITS + GALILEO_CNAV_PAGE_RESERVED_BITS
- constexpr int32_t HAS_MSG_NUMBER_MASK_IDS = 32
- constexpr int32_t HAS_MSG_NUMBER_GNSS_IDS = 16
- constexpr int32_t HAS_MSG_NUMBER_MESSAGE_IDS = 32
- constexpr int32_t HAS_MSG_NUMBER_SATELLITE_IDS = 40
- constexpr int32_t HAS_MSG_NUMBER_SIGNAL_MASKS = 16
- constexpr float HAS_MSG_DELTA_RADIAL_SCALE_FACTOR = 0.0025
- constexpr float HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR = 0.008
- constexpr float HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR = 0.008
- constexpr float HAS_MSG_DELTA_CLOCK_SCALE_FACTOR = 0.0025
- constexpr float HAS_MSG_CODE_BIAS_SCALE_FACTOR = 0.02
- constexpr float HAS_MSG_PHASE_BIAS_SCALE_FACTOR = 0.01
- constexpr uint16_t HAS_MSG_NUMBER_MAX_TOH = 3599
- constexpr uint8_t HAS_MSG_GPS_SYSTEM = 0
- constexpr uint8_t HAS_MSG_GALILEO_SYSTEM = 2
- constexpr uint8_t HAS_MSG_WRONG_SYSTEM = 255
- constexpr char GALILEO_CNAV_PREAMBLE [17] = "1011011101110000"
- const std::unordered_map< uint8_t, std::unordered_map< uint8_t, std::string > > HAS_SIGNAL_INDEX_TABLE
- const std::unordered_map< uint8_t, uint16_t > HAS_VALIDITY_INTERVALS

13.690.1 Detailed Description

Galileo CNAV message constants. Data from: Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

Author

Carles Fernandez-Prades, 2020-2022. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_CNAV.h.

13.691 Galileo_CNAV.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_CNAV.h
00003  * \brief Galileo CNAV message constants. Data from:
00004  * Galileo High Accuracy Service Signal-In-Space Interface Control Document
00005  * (HAS SIS ICD) Issue 1.0, May 2022
00006  * \author Carles Fernandez-Prades, 2020-2022. cfernandez(at)cttc.es
00007  *
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
00019
00020 #ifndef GNSS_SDR_GALILEO_CNAV_H
00021 #define GNSS_SDR_GALILEO_CNAV_H
00022
00023 #include <cstddef>
00024 #include <cstdint>
00025 #include <limits>
00026 #include <string>
00027 #include <unordered_map>
00028 #include <utility>
00029
00030 /** \addtogroup Core
00031  * \{ */
00032 /** \addtogroup System_Parameters
00033  * \{ */
00034
00035 // Galileo HAS message field lengths
00036 constexpr size_t HAS_MSG_NSYS_LENGTH = 4;                           // HAS SIS ICD 1.0 Table 15
00037 constexpr size_t HAS_MSG_ID_MASK_LENGTH = 4;                        // HAS SIS ICD 1.0 Table 16
00038 constexpr size_t HAS_MSG_SATELLITE_MASK_LENGTH = 40;               // HAS SIS ICD 1.0 Table 16
00039 constexpr size_t HAS_MSG_SIGNAL_MASK_LENGTH = 16;                  // HAS SIS ICD 1.0 Table 16
00040 constexpr size_t HAS_MSG_NAV_MESSAGE_LENGTH = 3;                   // HAS SIS ICD 1.0 Table 16
00041 constexpr size_t HAS_MSG_MASK_RESERVED_LENGTH = 6;                 // HAS SIS ICD 1.0 Table 15
00042 constexpr size_t HAS_MSG_VALIDITY_INDEX_LENGTH = 4;                // HAS SIS ICD 1.0 Table 22
00043 constexpr size_t HAS_MSG_IOD_GPS_LENGTH = 8;                       // HAS SIS ICD 1.0 Table 26
00044 constexpr size_t HAS_MSG_IOD_GAL_LENGTH = 10;                      // HAS SIS ICD 1.0 Table 26
00045 constexpr size_t HAS_MSG_DELTA_RADIAL_LENGTH = 13;                 // HAS SIS ICD 1.0 Table 25
00046 constexpr size_t HAS_MSG_DELTA_IN_TRACK_LENGTH = 12;               // HAS SIS ICD 1.0 Table 25
00047 constexpr size_t HAS_MSG_DELTA_CROSS_TRACK_LENGTH = 12;            // HAS SIS ICD 1.0 Table 25
00048 constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH = 2;        // HAS SIS ICD 1.0 Table 28
00049 constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH = 13;       // HAS SIS ICD 1.0 Table 31
00050 constexpr size_t HAS_MSG_NSYS_SUB_LENGTH = 4;                      // HAS SIS ICD 1.0 Table 32
00051 constexpr size_t HAS_MSG_ID_CLOCK_SUBSET_LENGTH = 4;               // HAS SIS ICD 1.0 Table 32
00052 constexpr size_t HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH = 2; // HAS SIS ICD 1.0 Table 33
00053 constexpr size_t HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH = 13; // HAS SIS ICD 1.0 Table 34
00054 constexpr size_t HAS_MSG_CODE_BIAS_LENGTH = 11;                    // HAS SIS ICD 1.0 Table 37
00055 constexpr size_t HAS_MSG_PHASE_BIAS_LENGTH = 11;                   // HAS SIS ICD 1.0 Table 40
00056 constexpr size_t HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH = 2; // HAS SIS ICD 1.0 Table 40
00057
00058 constexpr uint64_t MAX_SECONDS_REMEMBERING_MID = 150;        // HAS SIS ICD 1.0 Section 6.4.1 HAS
      Message Completion Time-out
00059 constexpr uint64_t MAX_SECONDS_REMEMBERING_MASK_IOD = 1800;  // HAS SIS ICD 1.0 Section 7.6.1
00060 constexpr uint32_t GALILEO_HAS_SECONDS_PER_HOUR = 3600U;
00061 constexpr uint32_t GALILEO_HAS_SECONDS_PER_WEEK = 604800U;
00062 constexpr uint32_t GALILEO_HAS_HALF_WEEK_SECONDS = GALILEO_HAS_SECONDS_PER_WEEK / 2U;
00063 constexpr uint32_t GALILEO_HAS_INVALID_WEEK = std::numeric_limits<uint32_t>::max();
00064 constexpr uint32_t GALILEO_HAS_INVALID_TOW = std::numeric_limits<uint32_t>::max();
```

```
00065
00066 // Galileo CNAV message structure
00067 constexpr int32_t GALILEO_CNAV_SYMBOLS_PER_PAGE = 1000;  // Total number of symbols per HAS page
      including the sync pattern
00068 constexpr int32_t GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS = 1000;
00069 constexpr int32_t GALILEO_CNAV_PAGE_MS = 1;                                  // Duration in ms of a CNAV
      page
00070 constexpr int32_t GALILEO_CNAV_INTERLEAVER_ROWS = 8;                        // HAS SIS ICD 1.0 Table 4
00071 constexpr int32_t GALILEO_CNAV_INTERLEAVER_COLS = 123;                      // HAS SIS ICD 1.0 Table 4
00072 constexpr int32_t GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND = 1000;           // bps
00073 constexpr int32_t GALILEO_CNAV_HAS_PAGE_DATA_BITS = 448;                    // HAS SIS ICD 1.0 Table 5
00074 constexpr int32_t GALILEO_CNAV_PAGE_RESERVED_BITS = 14;                     // HAS SIS ICD 1.0 Table 5
00075 constexpr int32_t GALILEO_CNAV_BYTES_FOR_CRC = 58;                          // ceil(462 / 8)
00076 constexpr int32_t GALILEO_CNAV_CRC_LENGTH = 24;                             // HAS SIS ICD 1.0 Table 5
00077 constexpr int32_t GALILEO_CNAV_MESSAGE_BITS_PER_PAGE = 424;                 // HAS SIS ICD 1.0 Table 6
00078 constexpr int32_t GALILEO_CNAV_PAGE_HEADER_BITS = 24;                       // HAS SIS ICD 1.0 Table 6
00079 constexpr int32_t GALILEO_CNAV_PREAMBLE_LENGTH_BITS = 16;                   // HAS SIS ICD 1.0 Table 5
00080 constexpr int32_t GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK = 255;  // HAS SIS ICD 1.0 Section 6.2
      Reed-Solomon Code
00081 constexpr int32_t GALILEO_CNAV_MT1_HEADER_BITS = 32;                       // HAS SIS ICD 1.0 Table 11
00082 constexpr int32_t GALILEO_CNAV_OCTETS_IN_SUBPAGE = 53;                     // HAS SIS ICD 1.0 Section 6.3
      HAS Encoding and Transmission
00083 constexpr int32_t GALILEO_CNAV_INFORMATION_VECTOR_LENGTH = 32;          // HAS SIS ICD 1.0 Section 6.2
      Reed-Solomon Code
00084
00085 constexpr int32_t GALILEO_CNAV_BITS_FOR_CRC = GALILEO_CNAV_HAS_PAGE_DATA_BITS +
      GALILEO_CNAV_PAGE_RESERVED_BITS;  // 462
00086
00087 constexpr int32_t HAS_MSG_NUMBER_MASK_IDS = 32;       // HAS SIS ICD 1.0 Table 13
00088 constexpr int32_t HAS_MSG_NUMBER_GNSS_IDS = 16;       // HAS SIS ICD 1.0 Table 18
00089 constexpr int32_t HAS_MSG_NUMBER_MESSAGE_IDS = 32;    // HAS SIS ICD 1.0 Table 8
00090 constexpr int32_t HAS_MSG_NUMBER_SATELLITE_IDS = 40;  // HAS SIS ICD 1.0 Table 19
00091 constexpr int32_t HAS_MSG_NUMBER_SIGNAL_MASKS = 16;   // HAS SIS ICD 1.0 Table 20
00092
00093 constexpr float HAS_MSG_DELTA_RADIAL_SCALE_FACTOR = 0.0025;      // HAS SIS ICD 1.0 Table 25
00094 constexpr float HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR = 0.008;     // HAS SIS ICD 1.0 Table 25
00095 constexpr float HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR = 0.008;  // HAS SIS ICD 1.0 Table 25
00096 constexpr float HAS_MSG_DELTA_CLOCK_SCALE_FACTOR = 0.0025;       // HAS SIS ICD 1.0 Table 31
00097 constexpr float HAS_MSG_CODE_BIAS_SCALE_FACTOR = 0.02;           // HAS SIS ICD 1.0 Table 37
00098 constexpr float HAS_MSG_PHASE_BIAS_SCALE_FACTOR = 0.01;          // HAS SIS ICD 1.0 Table 40
00099
00100 constexpr uint16_t HAS_MSG_NUMBER_MAX_TOH = 3599;  // HAS SIS ICD 1.0 Table 13
00101
00102 constexpr uint8_t HAS_MSG_GPS_SYSTEM = 0;      // HAS SIS ICD 1.0 Table 18
00103 constexpr uint8_t HAS_MSG_GALILEO_SYSTEM = 2;  // HAS SIS ICD 1.0 Table 18
00104 constexpr uint8_t HAS_MSG_WRONG_SYSTEM = 255;
00105
00106 constexpr char GALILEO_CNAV_PREAMBLE[17] = "1011011101110000";  // HAS SIS ICD 1.0 Section 2.3.1
00107
00108 // HAS SIS ICD 1.0 Table 7
00109 const std::pair<int32_t, int32_t> GALILEO_HAS_STATUS({1, 2});
00110 const std::pair<int32_t, int32_t> GALILEO_HAS_RESERVED({3, 2});
00111 const std::pair<int32_t, int32_t> GALILEO_HAS_MESSAGE_TYPE({5, 2});
00112 const std::pair<int32_t, int32_t> GALILEO_HAS_MESSAGE_ID({7, 5});
00113 const std::pair<int32_t, int32_t> GALILEO_HAS_MESSAGE_SIZE({12, 5});
00114 const std::pair<int32_t, int32_t> GALILEO_HAS_MESSAGE_PAGE_ID({17, 8});
00115
00116 // HAS SIS ICD 1.0 Table 12
00117 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_TOH({1, 12});
00118 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_MASK_FLAG({13, 1});
00119 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_ORBIT_CORRECTION_FLAG({14, 1});
00120 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_CLOCK_FULLSET_FLAG({15, 1});
00121 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_CLOCK_SUBSET_FLAG({16, 1});
00122 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_CODE_BIAS_FLAG({17, 1});
00123 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_PHASE_BIAS_FLAG({18, 1});
00124 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_RESERVED({19, 4});
00125 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_MASK_ID({23, 5});
00126 const std::pair<int32_t, int32_t> GALILEO_MT1_HEADER_IOD_SET_ID({28, 5});
00127
00128 // HAS SIS ICD v1.0 Table 20
00129 const std::unordered_map<uint8_t, std::unordered_map<uint8_t, std::string» HAS_SIGNAL_INDEX_TABLE = {
00130     {0, {
00131             {0, "L1 C/A"},
00132             {1, "Reserved"},
00133             {2, "Reserved"},
00134             {3, "L1C(D)"},
00135             {4, "L1C(P)"},
00136             {5, "L1C(D+P)"},
00137             {6, "L2 CM"},
00138             {7, "L2 CL"},
00139             {8, "L2 CM+CL"},
00140             {9, "L2 P"},
00141             {10, "Reserved"},
00142             {11, "L5 I"},
00143             {12, "L5 Q"},
00144             {13, "L5 I + L5 Q"},
00145             {14, "Reserved"},
```

```
00146             {15, "Reserved"},
00147         }},
00148     {2, {
00149             {0, "E1-B I/NAV OS"},
00150             {1, "E1-C"},
00151             {2, "E1-B + E1-C"},
00152             {3, "E5a-I F/NAV OS"},
00153             {4, "E5a-Q"},
00154             {5, "E5a-I+E5a-Q"},
00155             {6, "E5b-I I/NAV OS"},
00156             {7, "E5b-Q"},
00157             {8, "E5b-I+E5b-Q"},
00158             {9, "E5-I"},
00159             {10, "E5-Q"},
00160             {11, "E5-I + E5-Q"},
00161             {12, "E6-B C/NAV HAS"},
00162             {13, "E6-C"},
00163             {14, "E6-B + E6-C"},
00164             {15, "Reserved"},
00165         }}};
00166
00167 // HAS SIS ICD v1.0 Table 23
00168 const std::unordered_map<uint8_t, uint16_t> HAS_VALIDITY_INTERVALS = {
00169     {0, 5},
00170     {1, 10},
00171     {2, 15},
00172     {3, 20},
00173     {4, 30},
00174     {5, 60},
00175     {6, 90},
00176     {7, 120},
00177     {8, 180},
00178     {9, 240},
00179     {10, 300},
00180     {11, 600},
00181     {12, 900},
00182     {13, 1800},
00183     {14, 3600}};
00184
00185
00186 /** \} */
00187 /** \} */
00188 #endif  // GNSS_SDR_GALILEO_CNAV_H
```

13.692 galileo_cnav_message.h File Reference

Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

```
#include "Galileo_CNAV.h"
#include "galileo_has_page.h"
#include <bitset>
#include <cstdint>
#include <string>
#include <utility>
```

Classes

- class Galileo_Cnav_Message

  This class handles the Galileo CNAV Data message, as described in the Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

13.692.1 Detailed Description

Implementation of a Galileo CNAV Data message as described in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

Author

Carles Fernandez-Prades, 2020-2022 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_cnav_message.h.

13.693 galileo_cnav_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_cnav_message.h
00003  * \brief  Implementation of a Galileo CNAV Data message as described in
00004  * Galileo High Accuracy Service Signal-In-Space Interface Control Document
00005  * (HAS SIS ICD) Issue 1.0, May 2022
00006  * \author Carles Fernandez-Prades, 2020-2022 cfernandez(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GALILEO_CNAV_MESSAGE_H
00020 #define GNSS_SDR_GALILEO_CNAV_MESSAGE_H
00021
00022 #include "Galileo_CNAV.h"
00023 #include "galileo_has_page.h"
00024 #include <bitset>
00025 #include <cstdint>
00026 #include <string>
00027 #include <utility>
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup System_Parameters
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief This class handles the Galileo CNAV Data message, as described in the
00037  * Galileo High Accuracy Service Signal-In-Space Interface Control Document
00038  * (HAS SIS ICD) Issue 1.0, May 2022
00039  */
00040 class Galileo_Cnav_Message
00041 {
00042 public:
00043     Galileo_Cnav_Message() = default;
00044
00045     void read_HAS_page(const std::string& page_string);
00046
00047     inline bool is_HAS_in_test_mode() const
00048     {
00049         return d_test_mode;
00050     }
00051
00052     inline bool is_HAS_page_dummy() const
00053     {
00054         return d_page_dummy;
00055     }
00056
00057     inline bool have_new_HAS_page() const
00058     {
00059         return d_new_HAS_page;
00060     }
00061
00062     inline Galileo_HAS_page get_HAS_encoded_page() const
00063     {
00064         return has_page;
00065     }
00066
00067     inline bool get_flag_CRC_test() const
00068     {
00069         return d_flag_CRC_test;
00070     }
00071
00072     inline void set_time_stamp(uint64_t time_stamp)
00073     {
00074         has_page.time_stamp = time_stamp;
00075     }
00076
00077     inline void set_tow(uint32_t tow)
00078     {
00079         has_page.tow = tow;
00080     }
00081
00082     inline void set_week(uint32_t week)
00083     {
```

```
00084         has_page.week = week;
00085     }
00086
00087 private:
00088     uint8_t read_has_page_header_parameter(const std::bitset<GALILEO_CNAV_PAGE_HEADER_BITS>& bits,
      const std::pair<int32_t, int32_t>& parameter) const;
00089     bool CRC_test(const std::bitset<GALILEO_CNAV_BITS_FOR_CRC>& bits, uint32_t checksum) const;
00090     void read_HAS_page_header(const std::string& page_string);
00091
00092     Galileo_HAS_page has_page{};
00093
00094     uint8_t d_has_page_status{};
00095     uint8_t d_has_reserved{};
00096     uint8_t d_received_message_page_id{};
00097     uint8_t d_received_message_type{};
00098     uint8_t d_received_message_id{};
00099     uint8_t d_received_message_size{};
00100
00101     bool d_test_mode{};
00102     bool d_flag_CRC_test{};
00103     bool d_page_dummy{};
00104     bool d_new_HAS_page{};
00105 };
00106
00107
00108 /** \} */
00109 /** \} */
00110 #endif  // GNSS_SDR_GALILEO_CNAV_MESSAGE_H
```

13.694 Galileo_E1.h File Reference

Defines system parameters for Galileo E1 signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstddef>
#include <cstdint>
```

Variables

- constexpr double GALILEO_E1_FREQ_HZ = FREQ1

  Galileo E1 carrier frequency [Hz].
- constexpr double GALILEO_E1_CODE_CHIP_RATE_CPS = 1.023e6

  Galileo E1 code rate [chips/s].
- constexpr double GALILEO_E1_CODE_PERIOD_S = 0.004

  Galileo E1 code period [s].
- constexpr double GALILEO_E1_SUB_CARRIER_A_RATE_HZ = 1.023e6

  Galileo E1 sub-carrier 'a' rate [Hz].
- constexpr double GALILEO_E1_SUB_CARRIER_B_RATE_HZ = 6.138e6

  Galileo E1 sub-carrier 'b' rate [Hz].
- constexpr double GALILEO_E1_B_CODE_LENGTH_CHIPS = 4092.0

  Galileo E1-B code length [chips].
- constexpr double GALILEO_E1_B_SYMBOL_RATE_BPS = 250.0

  Galileo E1-B symbol rate [bits/second].
- constexpr uint32_t GALILEO_E1_CODE_PERIOD_MS = 4

  Galileo E1 code period [ms].
- constexpr int32_t GALILEO_E1_B_SAMPLES_PER_SYMBOL = 1

  (Galileo_E1_CODE_CHIP_RATE_HZ / Galileo_E1_B_CODE_LENGTH_CHIPS) / Galileo_E1_B_SYMBOL_RATE↩ _BPS
- constexpr int32_t GALILEO_E1_C_SECONDARY_CODE_LENGTH = 25

  Galileo E1-C secondary code length [chips].
- constexpr int32_t GALILEO_E1_NUMBER_OF_CODES = 50
- constexpr uint32_t GALILEO_E1_OPT_ACQ_FS_SPS = 2000000

Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.

- constexpr int32_t GALILEO_E1_HISTORY_DEEP = 100

  Observable history length for interpotalion.
- constexpr char GALILEO_E1_C_SECONDARY_CODE [26] = "00111000000010101101100 10"
- constexpr size_t GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH = 1023
- constexpr char GALILEO_E1_B_PRIMARY_CODE [GALILEO_E1_NUMBER_OF_CODES][1024]
- constexpr size_t GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH = 1023
- constexpr char GALILEO_E1_C_PRIMARY_CODE [GALILEO_E1_NUMBER_OF_CODES][1024]

13.694.1 Detailed Description

Defines system parameters for Galileo E1 signal and NAV data.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com

Mara Branzanti 2013. mara.branzanti(at)gmail.com

Javier Arribas 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_E1.h.

13.695 Galileo_E1.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_E1.h
00003  * \brief  Defines system parameters for Galileo E1 signal and NAV data
00004  * \author Luis Esteve, 2012. luis(at)epsilon-formacion.com
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  * \author Javier Arribas 2013. jarribas(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GALILEO_E1_H
00020 #define GNSS_SDR_GALILEO_E1_H
00021
00022 #include "MATH_CONSTANTS.h"
00023 #include "gnss_frequencies.h"
00024 #include <cstddef>  // for size_t
00025 #include <cstdint>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 // carrier and code frequencies
00034 constexpr double GALILEO_E1_FREQ_HZ = FREQ1;                  //!< Galileo E1 carrier frequency [Hz]
00035 constexpr double GALILEO_E1_CODE_CHIP_RATE_CPS = 1.023e6;     //!< Galileo E1 code rate [chips/s]
00036 constexpr double GALILEO_E1_CODE_PERIOD_S = 0.004;            //!< Galileo E1 code period [s]
00037 constexpr double GALILEO_E1_SUB_CARRIER_A_RATE_HZ = 1.023e6;  //!< Galileo E1 sub-carrier 'a' rate
      [Hz]
00038 constexpr double GALILEO_E1_SUB_CARRIER_B_RATE_HZ = 6.138e6;  //!< Galileo E1 sub-carrier 'b' rate
      [Hz]
00039 constexpr double GALILEO_E1_B_CODE_LENGTH_CHIPS = 4092.0;     //!< Galileo E1-B code length [chips]
00040 constexpr double GALILEO_E1_B_SYMBOL_RATE_BPS = 250.0;        //!< Galileo E1-B symbol rate
      [bits/second]
00041 constexpr uint32_t GALILEO_E1_CODE_PERIOD_MS = 4;             //!< Galileo E1 code period [ms]
00042 constexpr int32_t GALILEO_E1_B_SAMPLES_PER_SYMBOL = 1;        //!< (Galileo_E1_CODE_CHIP_RATE_HZ /
      Galileo_E1_B_CODE_LENGTH_CHIPS) / Galileo_E1_B_SYMBOL_RATE_BPS
00043 constexpr int32_t GALILEO_E1_C_SECONDARY_CODE_LENGTH = 25;    //!< Galileo E1-C secondary code length
      [chips]
```

```
00044 constexpr int32_t GALILEO_E1_NUMBER_OF_CODES = 50;
00045
00046
00047 // optimum parameters
00048 constexpr uint32_t GALILEO_E1_OPT_ACQ_FS_SPS = 2000000;  //!< Sampling frequency that maximizes the
      acquisition SNR while using a non-multiple of chip rate
00049
00050 constexpr int32_t GALILEO_E1_HISTORY_DEEP = 100;  //!< Observable history length for interpotalion
00051
00052 constexpr char GALILEO_E1_C_SECONDARY_CODE[26] = "00111000000010100110110010";
00053
00054 // Galileo E1 primary codes
00055 constexpr size_t GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH = 1023;
00056 constexpr char GALILEO_E1_B_PRIMARY_CODE[GALILEO_E1_NUMBER_OF_CODES][1024] = {
00057     "F5D710130573541B9DBD4FD9E9B20A0D59D144C54BC7935539D2E75810FB51E494093A0A19"
00058     "DD79C70C5A98E5657AA578097777E86BCC4651CC72F2F974DC766E07AEA3D0B557EF42FF57"
00059     "E6A58E805358CE9257669133B18F80FDBDFB38C5524C7FB1DE079842482990DF58F72321D9"
00060     "201F8979EAB159B2679C9E95AA6D53456C0DF75C2B4316D1E2309216882854253A1FA60CA2"
00061     "C94ECE013E2A8C943341E7D9E5A8464B3AD407E0AE465C3E3DD1BE60A8C3D50F831536401E"
00062     "776BE02A6042FC4A27AF653F0CFC4D4D013F115310788D68CAEAD3ECCCC5330587EB3C22A1"
00063     "459FC8E6FCCE9CDE849A5205E70C6D66D125814D698DD0EEBFEAE52CC65C5C84EEDF207379"
00064     "000E169D318426516AC5D1C31F2E18A65E07AE6E33FDD724B13098B3A444688389EFBBB5EE"
00065     "AB588742BB083B679D42FB26FF77919EAB21DE0389D9997498F967AE05AF0F4C7E177416E1"
00066     "8C4D5E6987ED3590690AD127D872F14A8F4903A12329732A9768F82F295BEE391879293E3A"
00067     "97D51435A7F03ED7FBE275F102A83202DC3DE94AF4C712E9D006D182693E9632933E6EB773"
00068     "880CF147B922E74539E4582F79E39723B4C80E42EDCE4C08A8D02221BAE6D17734817D5B53"
00069     "1C0D3C1AE723911F3FFF6AAC02E97FEA69E376AF4761E6451CA61FDB2F9187642EFCD63A09"
00070     "AAB680770C1593EEDD4FF4293BFFD6DD2C3367E85B14A654C834B6699421A",  // PRN 01
00071     "96B856A629F581D1344FEF597835FE60434625D077ECF0D95FBE1155EA0431979E5AFF544A"
00072     "F591A332FDAEF98AB1EDD847A73F3AF15AAEE7E9A05C9D82C59EC325EF4CF264B8ADF2A8E8"
00073     "BA459354CB4B415CC50BF239ADBC31B3A9C87B0843CF3B9E6D646BA43F866276B053826F3A"
00074     "2334CC5E2EFB9F8F195B382E75EEA63F58A06B3F82A3B5C77C1800FD9498F803E524435B32"
00075     "1210BB84690BED0BBBE16D363B3A90656A73720E27008852FB7DACC8284411B177728D9527"
00076     "C560859084A395A6F11A96AD9DB6B43E00642B000ED12BFD967868EAB1108552CD4FC89FBC"
00077     "408ACE7678C381EC91DD000319124EB5D5EF52C4CAC9AADEE2FA045C16CE492D7F43743CA7"
00078     "7924C78696FCBF2F9F7F36D8E623752200C6FCBBD71ABBB6877F3C5D6E6740AB0389458A6B"
00079     "66440858B2D383244E853646FE2714211DEA9E6196252815BB704A20BFE556AC474F899894"
00080     "4E0CABBBE21A6400B87BFDCF937D12B2821D59298AF4AD378F0F42BD8C41693B8D993CF37C"
00081     "8B478F3BB5D33AD2A9FA24AD7B8FA895FDBC04964192F7BA3FF74E0E3A435B5DFE042E3115"
00082     "CACF29624C0645E9C917534A2EBC1F5665E4E1B1BC56208DBCD8A27CCB6474D5D0E20CA407"
00083     "2C960E5ACE41BDA3770DF3B681F2B318F6F8E1CB17C2857350FB6009AED665E13B2780D792"
00084     "17F73FAC7A8A48048DB0FB8A8A5007CDDC9A7B2DA8257C99F1CB605A18204",  // PRN 02
00085     "E57DE19A3E4A8C122FCB1DD6584B3D2DAE364D800F9C5A9E957B38F624CBD3ACC58FA3ED07"
00086     "0B5E44857CCB813FBC0BB83B5D157C6C562422E5963CC4DD753C45B0264F8E136A0F1774D7"
00087     "7A543E44D51EF8C6B9408B6E3B5CEE1347A94F13ECDC94DC764976E5A50B4CB0AE7557553B"
00088     "47EDFE03EC2CD32EA8D125A341E1EDFC77E75330D6E7B23DC838EBCE7E5567F5B8C80C3D15"
00089     "E7404B4E10F0BEB0C69626A814AF91334199864FC77E0FF548DC2A6FA6A71C3C0561F2B085"
00090     "CC05E8512E27B9DBA60B93D114B87935776C8E9A67905C429D48BF3AB1B0A56FAFBFD5D9C8"
00091     "D8C8A9E5918BFF273CF5E8664FF2B90314BDBFDAD5AB8C22A0E45C104ECE75EA43FE9BDCE3"
00092     "06A5A28AE464628163D249D8056005F1A900951808CC8620F81768153436F741667A8E271D"
00093     "D986C7A1E5046FCC74C7CEBBF9A1296D6CF0B2FF85BE412D87214BB368DFF462AD649D7324"
00094     "A117252311C664D33E4DAFBD830FBCEB6EFBDD7391D4BADA7A775FD1949D981F619655DB3C"
00095     "22BAC34E5AE41222905C0C7E80D6EA28471EC0468756531C09A471EDBE200472E78F1701FE"
00096     "E96E5769A9893C0F11E7906B064442E06E21ED8B0D70AF288690C532A2D03B373E1E0085F6"
00097     "2F7AAA658B569C5184E3DDC40ECAA88B871186O1691892F9F55E2DE79E49DFF11D434C2BA"
00098     "3AA6447522A7C99DC215CAD2ED0114ED62CBDAE9D315E48AE14D2014B7F8E",  // PRN 03
00099     "C0FC4C72A12023BA7093C86775DF3D2F42C7CEDE616876340BE4301361B9DC9DFF4F1DEC6A"
00100     "62E165927BDE4F809E969AAD085437496BB95904719820F4CA8ABBA0B84C34B06DD7E268BA"
00101     "10E386FA7DB9FCFCDAF2B6AFBA46A8A299153B4E11582FBA7F28F0A0F9DE41830AB3333506"
00102     "2C57D81DC361EDFE491939100FC827F36273760043D1C35B74E36C6C4DBE1D307847D55AC0"
00103     "7D8B212C2DBA632A86AB15BD0FAFFA43070644C7E50623195A3796AA8E8D6E4E964FA0E448"
00104     "8A500B9063FBBFB1204A0E33C6CF2879AC2BA7C86CAB57E3E8A497836194E65C5C39B950F1"
00105     "AFC3B58E850A5EC39F4190D55351D16529CD52B36DF4A2DC68EE202BB758CF19C54B0E1461"
00106     "D547B5D06C2F9DC09C2B15458C3140860E4C6F3FE4F417FDFCEDE00F71212EE137E6669E56"
00107     "9A7845470CA564F85CB4772808D65D2B48D409B709BD7AC5F7E28AA804CE9DAC3ABB5A5B76"
00108     "8C6A184B5A974E933F2C1772FF64AB26BA2D5A165744E314EFB2238AC4858A8B82723DAE88"
00109     "65478EAA261F35DD4D98A9C07ACB0B822AFF1AD3E739CB214CE737196FEF2DD0B0D45BAC42"
00110     "3935670BCF71C2EC04CCB98943786173C309E75A02BB78A788A5E6F8A8F407E57B8403841A"
00111     "9E1FCB3A7AB80D1F6529770E52C173E2C47EDED4400D5E665E325ED845C9E8D0E66FDA16B1"
00112     "7D61EDBB336F22688C3F0FB040A55F33B65FA9F3D45F5B22C445CBF9DEB22",  // PRN 04
00113     "EA9596357B343DFC31D5875CC0E94117A33651472E476D3892D8112EB6CB6E0151D409C5A5"
00114     "14DCDA38A773C58F18B590EF9017B6EDF0192AB7EB29DD6E1E7E7390C13E9B10209D5775F3"
00115     "B066F7B2DBB7307FB44F726DD2F368A5FDBE75BA7248762E1EC7E4589DF1A353A16D6B3CAC"
00116     "1C9ACDB89890ED2C4F44AFEFC763DB51D102230C37E1ED0943CD6F4176B2F5C19119588911"
00117     "ACF81A7A29320AD579C1BFAED1A70DEE1B87037138ADE411E0BB92F5B3148DFA11F2F84CA6"
00118     "C019124B922837503AA9823A97E443A66378D5CB3130A7EC9B05670E85D095D5E6F603092C"
00119     "632E51FD9013FE7FB9F08448FD09F1219A4744CDAF82BF9C60039C8185C7E9559FCE301C6D"
00120     "3F46A2E514AAD44D3889C8CB4ED7439BF47019194F2644363770F8BBD0AE92B6F5F43CBBB5"
00121     "03A885239DA636903D4C264B3FF09AB77E3FDBA7EFC63E0792B6D5183759E57D8A694CDB13"
00122     "3B4A9E301CEEEB978050AD9A9E410091AD29E389829E2F24BE1E3B24F4540C4A6533EBA72E"
00123     "8AD540BAAE43A0CB82F971F3A51DD77FE9E1956E2EE7553E050A1D10B99552DDD5B68F2E28"
00124     "59712835BD2AD6B08881753B4833FB04740E3364D2CD4921B939393E7EA91B854FA1E5A8EE"
00125     "79FF0A83F111F78435481D462E0E1CBC0C921D190A435A1BA755E4B7021244FC5E3F0630F2"
00126     "A1F439C02AE619393E5624834B05ED7DEDE5F0AFC7A40899424E75D4EE792",  // PRN 05
00127     "90E92279CD4F60D98F6E8FCB3E9263DB60FAB146A835AAC2E96B3BE3FF07119032DEE0521C"
00128     "731117E90C2943B389DD6B65C5E21C34F86F5A7ADE04072DFD1479EA36528D340736B0FED4"
00129     "F6207BE9F6CFC971D5EA11781AC2DA25DBEEB6B903EF8BB0AC0CD2E29F94B8CB67874A7B74"
```

```
00130     "41045758E09EA061181A50E0AB7BCCF801554E0644780BC137436E3FB7784C182856A790D6"
00131     "943BB53DB40D13D6A2F7B83A5C521073883B90FB8DB1C0F954D132943C09156A09984B8220"
00132     "79FB8FD09BC07C1D6336C7CEAE8CC3162760B9838CA6A38FD0044FDF099E416D57BF9F33A5"
00133     "51043F34EBF9BAA90901E62D2D981065F977852072F692535DDE24EE8946387B4E5B0FEFEB"
00134     "D75552C1FC325A608A78079A9AC864F2F30010A3304CB16A26AF98D9BFD3B8D128541190B2"
00135     "BBEE275A6F53B9BC5108306985ECBB983B56E34F18B48A12AEAB88271F4F780CFDFA83E05E"
00136     "35C12464F4350597CCAE9B4498F5A5454DCC3218D3336763674934ADCBCB5EA52891EB240C"
00137     "362248226DE64899BE30735F6495E94AA61ABEF62B803C57FDD045B724ED1966B6E7DFDFCA"
00138     "5B36F7B0FACEDAC62DE8E10B12DFC84B1A9CEB407BDE63CDB5208ABBE5E066AAF262187E94"
00139     "502B1701B2CC8681CB616773DA2B7AF49443CFF528F45DD7F259598367719 08C2519171CAE"
00140     "D2BCDCFCEA46301E7D99A5AF7199155772E92BAD85F35EDB656F0999EE828",  // PRN 06
00141     "A91F570102961D62CA6CB55144AFCCEAF3910F3336DCB029CDCBA164ADA72732771B6ECD1C"
00142     "58E49F468A2BFD23E1B996DABABBAF5AB3A4C74926187B5833006F8BEF7F9CD0F05A2A0B9B"
00143     "D9073C4C3976E8660CE7BF81634CF0B31C3DDD806A6A0C15BC552B83A86789CC675A6D137B"
00144     "E27BC86DF68FEC5D268119EB9E965260FE1F5C56AEF60A8622CDA8C42F24CBA7F5B07A7416"
00145     "917277323314AFD3ECD10F74BEE7B22DC760EFA7F935FC9963411353782547FAEED32E69A4"
00146     "FB5756C1A73CCDFFEDE50F4B2D9B5D2ED5C59C9A52D80CD27B989B8DAA14C569E763C08FD4"
00147     "2358CD064B2DE0526607C9536D75E1617EC80615EF5EE2314FAC29907B61B61F8696CB80B1"
00148     "4B3A0148EEBC825C91150A08A23FC7B38B5982AA02A18BF6E91B3A1F2EEF360F682A34AB36"
00149     "CAFCAD556841073F219910F7BC2F07CE45E98F77F50475DF9EDFE2DC9E3D7280193D61AB50"
00150     "76A14887E9D9193C3B83C5773BDECA067CA1BC3D4561C3A8B4E30072A6269B529760CA1B5F"
00151     "E9D3DB2B5D1202CE8B18E9E2E80FAFF47108168D3C7EB3C940B1A35A1D1B968A5A9DC0686D"
00152     "D8336E498C240F20871600FF995B9E33169DCFCFCB58E75C94D82F843C60A7118F0D7B4006"
00153     "4A8A4176C5158E86AF0BE4C1D5D73D1C051132A85CC0628486AFD660502A515D6353B674B1"
00154     "D4E61750C13E8A3AD48FE1F89F201C288A8F443867C2BAC23C706EE7A2D2C",  // PRN 07
00155     "C6E00978E351164532EEA256ECBE0D4F8FCE02A276BD19666DE93936F7A242FC4C7E879791"
00156     "314B043ABF1D5F9B0036ED22AA92028C800C4D62BD6640431170EA77311865074D670AF284"
00157     "7AA47CB94584A793FA82F51574BD7C62BF14386F14A3D7DBD129FDE64EAD67EB35D5E13FF2"
00158     "14D7D163B770D4A77A62D02D88C0FCF3FA5EC306EB7F85539105FA2CE5F53D182E58FBBC1C"
00159     "57CFBCD2D2F7FC8A067D6FA0BC834DAB8F370B0971BF6D068CD4D3A32C11C6598DEBBAEA04"
00160     "6528C5EF762828CC84D003847069FA18743A809A004431E83924B8FDF0AC78699B905ACCFF"
00161     "82E83FDAFEC8648DF64042FC9438B261B73F0541498ACAD67D702AB631BECEF8680D33CE8F"
00162     "4F0CE29B95132591A350DD68B36734B97D4B3E84A76497F702312F2A8370DCF26A7C3C8EB9"
00163     "1DD8699C48F551750712683E0397083714A6CAC3457C0FA70BB3A036C6E0BEF24E6B20BA55"
00164     "65B351C2EFD56BD9455FF7728BE07A097208E73DE4CD0CB4E215B4642365123CDEA419B284"
00165     "59D50E864B762554E7C1D7CAF73DA7D40EDEF5D824A2FE1A6CA473B07370932A8A5D441DEE"
00166     "3C9A60DB68E27A9D3E9C8229B44E5B434C6D18A8CADB6D17BC4614DEBEAD670C73132CE2F9"
00167     "99C8716D1098C69277E8ECAC546EF8002E5182E25F31A354DF112E97F8733DD20893B430CD"
00168     "7130E69ED4A0FE4D6C2E4FA479001E42EBC9F36E5DFD3E0BE35A64B89745E",  // PRN 08
00169     "821BBB3FB91E50253A9E71AC379ED57AEF394C2CC59587B2D0337CE74002EEAD17AB5D504B"
00170     "CA68BDAE9061C3DBAE2985EBE292B9BEC9D3542015225F44ED3C2C3FFB036A515BF33DA169"
00171     "0F3438FD225A5034106C5F4BCC43301EEC2245D73F63038E2A7D9B8CF95A9FD813FFA071FF"
00172     "DE423E0CE737969578BEB909764A8D6DAA9E15A4FA0867831652C0F6E9AAA39A63F0AEEF62"
00173     "A433476CC7380460ECFB8B7F3B2FE8C4C42A3EF1CDB808FC9747FB4F044B3B47A4EDFCC946"
00174     "3ABB72C55399B2F79EE5FEDA270D6358B27F8466969DE4A5F2E6A5F2C4CF0813C09F468DC9"
00175     "7FC0E5DD057A8A0355767B698F8A79BF0350C4200413A15E6591DE70A1B502E19FF515C3DF"
00176     "36935974A4764895B9E3CA2626BD39B7ADB780AAF7E2E914E804CA923089A51F3876649C73"
00177     "CA3C2623A8C95D11EF4B3F941E9772EBA1F47212C666F03F01509FF699F74EDE27182B6E98"
00178     "AF49D1BAACB41A328A8C34D6E8AA3553DA3962B27B041495F269328B6BFB4A385CBB118953"
00179     "F3F009920EC4C8590003290DD60AC89177BB8C4BF753CE723AECA392B8D9E5E9E4113DD062"
00180     "F294A77B6EA9A0477E697C04C787CE78A92C704409D37D37B6B392128698D0D8D4CA101EB3"
00181     "8B92F467F0D86EFD8759A14162CAB55F8C457E82392790A5BDDC8DD2663944F880C95EC02F"
00182     "E5363B064623994EE5D4396C0E44DE2A3D225830BA6160270BCD110A942B0",  // PRN 09
00183     "92A0DEABA9875D4AFAF99A24C1D5F10EBBE6DEF9CAE5B0C85B2A0417C1CC5D1A5F71CD8F8A"
00184     "4B013C3F012C0A19EE4A23106CAB8662C5A2A93A971D0B6E487FC05BAF5C355A9520C91485"
00185     "84CFED3EDD0F38696E161E64378C831C586D9178A0CE289A67F33AE68C02A3CD138FA09DF1"
00186     "CAD01EFADFC8BF6F5407B79B18D09C82804736752D08A1FE09EB35F544E9F797EA36DB493B"
00187     "A947AA82513EB1615A356B5AA4308B0B4183E070EB494D628159D2D4BC3CB110AB0CCB2E9E"
00188     "73B5B7EB567187621E72D99F1FB78565917B28464A5F29DD8D6F98B6ED703040A44B0ACD97"
00189     "F15049E009E8533FDB0B6DB2F2582E6BBF81D7B0EADC8F402508F6B8531AD13FD1C55978A8"
00190     "A70DF4E053DD475132D348AE27581370EC14A3E0F96E0D70DA4946DEEC0760011404FDC5B4"
00191     "36CA7419D05895F5E0EAEEBC88C74947733BE9919F18CE702887A6C4DF7C19279B82FB6460"
00192     "90822DA9CD9C7653F6B931A337A28F7A4A01DE0CC0744F22961045F8EF8D4B30B07E5EDF5F"
00193     "A944EDCFB9841A9088AE82444FCB6E90B0E9C567A80E8C42EC713D78132F37AD1D2592C31C"
00194     "93D2EAEFF38AD94E5C0D94F949F47B88B03BC1EA4E5EC9C7D9DF19ED208B8E44FFDEB0B625"
00195     "F633C7DB1C826AA9E1C1309E5B14A0DDDB79714DFDCB52221CEAD7E8A140DF7806F1271564"
00196     "78AFBEE922B8ECF322D66B48BEC434299BBB36B3BD9030467B7F2EBBDF358",  // PRN 10
00197     "AFA7FBAC93326D0C36A388831B99DF4D527BCE7C9070F7B46B5FFCDEB07384801AE5F86A89"
00198     "934DE23DFE2C1AD117797D4FA1BBA6175823B41166DBE9D126F17B3761E2C352AB396A5A9C"
00199     "CEA42A5E9EA1BE3497C0A5BA9121DB97F64159AAC78E62D7DEFF3BF4CF73F8CFBE045C9D39"
00200     "E41D5D208DCC4B47CA27E900C3CD8FD1408DC5E0F5114F2FE65817D37CD1452C4967ACAA21"
00201     "19FB8D60E5E2FD8A820D0AADD88B94D40435C095568AE6394D3B97C835BA868A83083316C4"
00202     "9C75D36EFDD85165BE74A4F2B2D21295EBCE085D9C4A4758FDD9CF71B97FDF34B7B63A5E96"
00203     "91DBDAB834D87D5B52CA9A53032FFE821398616EA92625C2DB633E37911987083A3B49A86F"
00204     "C562FB1264A75643A5FB6E97162E16ACCE353227FE61A859E094C2359BC4645946AD12AE5C"
00205     "39C70F59EA7B597A9B3372C23AA578146781A61163C92816627DD9C4BF1788087821F9F5D4"
00206     "1B75A0F251B06BBD3E29ABD41E72A1D48323D24E2AD6F11C2D49678CC04FCF6B0EFD33BE6D"
00207     "DCD444F5CA02FE158112631F782CA7B0C5F3607ED807495BF8E82C5EA51A922FE28C8168D9"
00208     "844859E7A3EE3038C5D1D4BB4B13406C340894DF46406836739E31D01082BC84489592DA0E"
00209     "985630CEC40702A36DDC301B3AE1E8101786FEDBF752F9E175287C239C18FC25795BCB479D"
00210     "EF59C58C373313C02A1BC5F16355E2B50EFB588556708687 28B902653ED80",  // PRN 11
00211     "943CAEB680AA3E630755DF32F406F403D7AF5E48A710274D3887A7AAC8EA6744B889F2E0CD"
00212     "2033DEC0B434A9591254A0AA68C5C9BF11D35765E86B437497D84E5DCBBC0C0C580CE9BC50"
00213     "EC6382AD74DB02C2C233B7BB07517D48056226C505ABF2DD244F6BBAA23313D570558B065E"
00214     "423277680 78EFDB53DC465DA038E3B216D990EE951B3E13D3C1CD559998F77BCDCD2B9522B"
00215     "6F1DC5E12C912EAEF574AFD69C251F9B2532501AB9F4B3B2223D0F8920BD562B0D358A14AB"
00216     "0D196DF6337D1C96CDB47AFEC6F81DED4B5773864DA32FCCD06B9AC53C122B2C6327E6E5EF"
```

```
00217     "E227DE4893FF15BBB2257FAEA836E99676EE32BF6FC14D4F56EA191B8A3870374A0867C49E"
00218     "B0015D1C6D07B87A36BFDD1DCEF20EA7B80D997CBE2D83EB5630F2EE6F73B0D50700C89E4F"
00219     "32438F5541360683DF11DA6E7A3C1E7DB2A87800D9245BF04278C990A8DC9CD86DEF39CBC6"
00220     "D4BC00FF13BBE132F9D86681A8913BE787CFC69C35304824788716D52DC74CEA399E06DE62"
00221     "41780447C74DA8E947134D8B2FAA9648D6D5F34C9D60AE5973B5BB0187796D589C8FDDD767"
00222     "5671F28C04AC1038D09251980683CB712F694D7C5B0D5B1DE86CD10EAC4EA04A55BA8803D7"
00223     "8249BEF516D38067890105A23212E72879FA267A8B4F0455A81F17CFD3E5DDC55E5D4FE00F"
00224     "83E18626C676DAF00E6AAFCC23D209DEE0B0FC6C2AE4DE161D13017ADB5D8",  // PRN 12
00225     "E5E70E7837D094416558C044D758383EDF5755C80921218ABE76E51FB93249E211A38FE6D0"
00226     "7A7DFD2263E6E3D8DA0F921A06A606B804DE7AC3FD097E5F96EFCC0F544D623FD6F43FB88C"
00227     "EA7C341E901CD47A7E24AB141E998FE41CA87CD6CE8C1870D9ABB6503BF7E8B659084BAF22"
00228     "37DFC94F35C9884C7F44B87120BFCB298696E613C1656AC4899781A94869EC603B4D386653"
00229     "37CA8593AAC83AD8BECE10302E4B4694237E96CCD3AD9CD5F8EC039A1D1A4210716371404C"
00230     "5C3FF375CB3A33559B1C1A239F2E442C8EB033501BB290434BE73489F7169653939894422CF"
00231     "4D57E5B4F3C76AF3C5E8999E61805134B9D7C40BFB59D0D0FD30F98567E66D6148D6AA64F7"
00232     "4A22C50AE49D6B1ECC6BB5A002ABF38FF2E2436766B86BDDE7D95DD6E02AB0FF06E7BC22CE"
00233     "C98D55AA2BC4D7B91C36B2FF9F525A74423498D548318509320FCCBCA582A6C2996AF65384"
00234     "22FF0DF060C0BC7356B0850A139AC3914338127B786F4BC58CEB6064DA881376A147DFF53C"
00235     "6700BD13316A5874A75D7B9713DF54FBB393BAFAAD7F7B0710C049A0B6A8B76A9956BF6185"
00236     "BA39D9C347D179FBB97D4FED68F47DB5AC8E0D40122EA51C4A1F88D23153DF651A180C2AD4"
00237     "56ABD7F851B65B220A72BA48FAD0436332E4EE7EDC554B7D75481EE05C3D3453D760E9099D"
00238     "D27B324DD84C0C0C4DEC4C674D25284B16410F959FBD09D9DF09CE875601E",  // PRN 13
00239     "BFDBC82ACB4FBCD5A90C5967EB2FED597A02607F426002128AF4B38942C85AF4472B3CBF3B"
00240     "183F240E049B251713740A31117F108936631FD0F11C5F79325BD6677A2C2B242965AEFC14"
00241     "7D93358730AA782491209CBE600976F56030753CC979C240A196647CD9EAB1DD0380E59BC7"
00242     "905EF740C3411AD9DD72027D0D3DD6DEB0F5F3C18F6D6F7BC59B758E7E262937B4599B3856"
00243     "7C147ED2689BA2CF23736CAF55B6925827E2B70E47D3813C94C85298BD6B49C97B5D0221BE"
00244     "9E3164B6FA3D95AECF53AF170966090F19A69E75F188BD2556B4E8FA7DC4AC6C34F54297C0"
00245     "6C2A96DD1C45B42E6175B5E8784568F7FEF0B6C124C5019CB577B374941E8515CCFC21F46D"
00246     "188BDD2C2284C688879A5BEC50CCB97FAEE1F75580577498D509D3DE161BE216C873B29E17"
00247     "8CE17DCACC5E9E2224D05ECC842FBEAB82A75AAA20769FD81131CFB69D5E35409273CA106F"
00248     "FB27F63FF997CB500F161F6DD3A8BFA5719F004EC17860152D3290951678A131E4F3D3AB34"
00249     "CFFCAB2967ED9D8F1BB987950306BD28751D2AEAB05F071B08574EFCA01E5386E04F727BF4"
00250     "13A8279E9392EFB64D9AEE00877C76C81EBC861E2B484A2D35E592A131726CAE61BC010B95"
00251     "4721A82C968CC6F384D9BBB99B4E87846D10B94EE31F64846A5834DF73A67A267B894B1C06"
00252     "242D750F15F3E1E850A11CB2E2B16155008F91493AB3BC77CF9BE56F9DB20",  // PRN 14
00253     "D64F3D1CB54CDB9143D9E701BD313779C09DA064D9A85674CCB53B0C5B4446C122098961D5"
00254     "EFFD6A85537486D5EB26B5E18FFBFB8E6EF16C2DD2C02EC7C07DB15CE33015A636E225F744"
00255     "C963BF0653A89A48F1AF04819E273A3AE1F5538AD574D553C5A0DEF47B552957037BCA9219"
00256     "70C76DDEF74BA083ED55363760A6780612C075964B083B4F674EA0012FD1DF09F0445CE75A"
00257     "698852098206868AD8241E3B319FA8D2D86DE6E7631DF1AEB571F9676323E0627307F6D8F5"
00258     "69536A758DE5EDAAEDF80F4335E3AFCAD07F70AAD5CD08CCA1E71B84D4D97931F924AC0010"
00259     "C0811972ACAA414B89FFF7917E653BB31E9CDFC72595066C662CDB9BBC96152D46BF4E8C15"
00260     "A8D34809C4B9D79871BDF0B63FA294F2D667624F6E0210CD40C92F1C033C3D8BF089EF85C4"
00261     "F571CA727C71B23128A9B0FFD70CEA93C316FC4D69D79B089107F292E03425B2552AF5AA18"
00262     "FDB9AF86EA1972B66B1276B09119437E4DFB8F8E3972D91A93816EBD7D8D715CB47EFA7429"
00263     "38B0B49FA27A291B0DEA1DF0B8F878332103F45A99936896181E51FF65C6995F57C2C54B80"
00264     "02DEFF54B0EB3131EE7D61030C33B5502C49CF398FEC4B7615D16FCEA3E8EA12BFB311D426"
00265     "331A06606CA5A066707C4AF8D1048F1CA6065FBE506D06C6C00D5D250E2272655518674681"
00266     "6F05155FCBDE24D4AD115BDA98AFE08B12A1F32E7C2ADA801FFB78BA05726",  // PRN 15
00267     "9D6AD9889EA02FC9A58949290975DB0F512EB37C8156CC9F1242B9E45F22CC1D6ED1CBCB6C"
00268     "B245811CE729261641FDF7A8F389BAFD7311B8BD689E02409F6E8C5202F466349EA466E539"
00269     "8B29C8CB126D9600D89697A07A6900FE8D95951903DAA3419839C2D9E35E9F4EABC04C9006"
00270     "EA585F544C7163A33D7E78DE28256B7B8978FE018CB529F7F79BBF66DC4F0DECE80AE3C2CD"
00271     "479D78C4480E4DE2F06C70E5FEBDFB4ECAEDC2E7BD891AD6C91A7C2446F1B13B340B716078"
00272     "2F6CC5B45F9787CF1B0985202DDF02EC552A6DC41325FD8D31A4316C13C56F7157134F66E1"
00273     "D103CC3AA7EB951C92094EB4409E6E7BC494434FAD80999D46D824A5A57390599052025F7D"
00274     "A4838F7D16A8DACDAFA06D175546FADD1E3F7975265230F6C01B9C1FB1B7AB1F2FDD43A577"
00275     "8E3C88FBEA70575CA26D94D249670E4D9FF28EC67D15829776D7BC6754D2A2BB01554E5FF0"
00276     "C3FAD8A1CB546E8AD5E5314103D086D14ABD30EA95DDC591C13D96C1CC3F60FD18D216B671"
00277     "81B6324AC09A97C0C45E50EE8380ED42F6E0430639373E7760C708248EE7D74830E9594114"
00278     "879748883F247D056B2BA94A0FC54CECF6F5C6AB4DCB7CFC8C224F40D886427504233DDBED"
00279     "CE160DEFDFFD69EE2B75746D9CF71676DC453FD01C315ACA96373ED387B040BDEBA7FF3CE0"
00280     "0D915F90AE6E1796971F8052160154E8986913AD7BA291188EC49A60BE27C",  // PRN 16
00281     "B5184F7D580935ACFF18201CE8B5D54CD0A1CACF102FBC8AADF391C4CA5807BAEEF4E5E47F"
00282     "7459E74485E48E0C42D27CADE6970714FD97C08F9592FDD387C859FC12C1CCCFC3EBF510D6"
00283     "6FBD8C448C25A322CC5887F94A55D48ECA362C690F24833C3B032A047D12BDA2ADC6824A1F"
00284     "6EA9320BED27968E9CFBDEC60D041EF538F1740C0519003FAA89CD4224293167E05344998F"
00285     "D396EEF618E8F547990BC06A8B76D0FD6FAC13284601AB7191CEB813C46C45CE7B3FC09EDF"
00286     "08DAFE136BFBDD63E6CE7E4BCBB16C5DA68AC71A1298FD27363349A261C2F2CA8CB799E860"
00287     "4ADF70092BDBD6A04CB80568776A537AD1711891B251C74E42FCB095B23EEF70F167E8B485"
00288     "6BB7F92E3A43C79FF4437262DD70BAF9B16CBF5F10D1AD7559AB0F8CEE1B9FAD058E84FCC3"
00289     "42D9F0D9FBE4207D40E281416506242CA1B8DAB28DE88D2D00BA21AA7FDDC25940CB29F028"
00290     "11F8DC6850A6A87D72CA9F3476A73649FB4A254B1204CC1261E7D512BFE7B0D0091AD5CB0F"
00291     "BBB765FB5AFDFAB0D701941DA54832FE8253BC0CF61924BCA2CA231A196C7C32A350AC9A5F"
00292     "A2884D8571FEEEDB7D29632E71898BB62B5E4E0104F73AA6A9C6B8CDA816872805D75ECA64"
00293     "F961641077B259C9D39E2F3CCD9FCFB1E6B6E2692EA34336A967E587F32E49B961B9131119"
00294     "8A204D11874B4BEBC6C04DDB5B82D5B741D3CEDC03A56A2017B3D2C4FBBD4",  // PRN 17
00295     "CFDD6B78AEB21CDCD6AF8C349F6DF8FF8B96BC8246A672A16E45B5D0AB7D992570EC45A534"
00296     "B77F204039FE200D4C5E7C78FE24941F578097B216177D8AD4E1844B2E52D843256D0BE850"
00297     "4CF2D5B639E2CD501A6FE39B8AA7DB7DEA924B38692E43195DB7E5F25E25152DF0FB7E0D4E"
00298     "F63F99CD95F699E16576702B651C29583645070011B2A1F88C947BAE7C94D48EB07A132DB3"
00299     "8D4FE2B77EEAFB31AFB442710BD0AE4E6102DA69A454517B6F148D97DBFBAC7305979B5D74"
00300     "D7D7568A0CA56CA89F23D8330261025CC741F9D7A4BDB356B544C68C89CCC2C125F5C71E18"
00301     "C4EA102343AE4A44F6FC695810E6F28C86BF53F4C8B8AAE46DF6006B1679EBEA790266D4D0"
00302     "2A2095074ADA634EE60C7070285C316E1F191BC5A88B80D673F144D65B870A65FC93D8B4BB"
00303     "29B80FD58F9FE95F5994878308CAC5394781E4D5A3F5EA2A8ED834EE5BD31D2058C843F22E"
```

```
00304     "B778C4C25144193DAA65F9B57AEC4A344713E9EDF913F3CD29196B42E71BB182AC3B1A60AF"
00305     "DBF1112A86A20BFC1D28D3E0DBBABF38E8F12651C207C951654FE8C4CECB6C6F93EC46456D"
00306     "AFFD7320DEC8D08F2F712CEB4D82407D61CC47B333F69310C06EE1FB5ED84F83945F05D4A8"
00307     "7CF5A68D78B5536880DE3443E804040E599BC5837E22150C93CC1E5E711F9B889C78C6FF88"
00308     "2D80857EF41ABC5F12E99105E6C894EC0B796E0A645780341CBD039E8C6EE",  // PRN 18
00309     "ABA759AE16B9D8778FAC203FADF48015331D6499B8CD74BD71ABEBD3E53ED90625E3057EA4"
00310     "7BE587600F308D38743A686EF6FA189A4D86E4A35EB798FD2307345FBD10FA701265F64176"
00311     "03365FCC4CE7635924428167115BA372294C27A23CE6C27C506603C5A6618A2B3344BAC50A"
00312     "B7FDC29D36BCBDFCE0D48D088EFD8EA1DE492C543093C30AB7694627C01B334CE3368AEB4B"
00313     "B3267EBB1096450BDFC2571977D7EF78D6E288FCE0388A041838EC2031248F5FD659C70180"
00314     "634A1DC7196C8D9111C75B51C50F854CEC63DEBF9FFE1AB9406735EC3187276DE7CA2FAD42"
00315     "87027956C93B8E84B7C0C3A9C3F7E82B3DB35EB6D2CEBDFE0708FEDD764C839954F2CC9044"
00316     "B652D0A01D28BD6B9D3DD9740CAE39AA52597FFC1227FAD8B78EAFFC31BE94A632A1AA7A60"
00317     "AA5A9E090DA2B62F6DBDFDC50DF6EBE1D9949619FE9B2302248D6C801DD2D6C01FF8206A93"
00318     "C0AD22C6990C4EECA7D4BDF36C3246A5D2D2B3982C608E6AD6BDD85C92682EBDC9E4117F8B"
00319     "7F841239C2A5AD7977E11E4E9CA73A55859EADF7C9C2F1B28A6B4AC7202019230063331FC5"
00320     "586756CEA1F8478173A0A4964D00C1AC099590152125A4D01592C54DC2555E1BA34C7AC039"
00321     "394D1979AEA2BF7B2B2A8CB9D62E89132CE9E3B325F023AC6E8117CE57AD4B271EFB0C172F"
00322     "BFF8FA6A17A490B67CA7B15F865A8AEEF37651A622390E82AFD418C7AFD48",  // PRN 19
00323     "CEA29601B96AD3A831646922000BBFF02C014A9136D9A151A0E61A51F9FC2EC0C3A8F4C83E"
00324     "64BDE569A33B4CD653C1345B7CBEA3B3AC0411B6145727B1DBF6066ABCE9DAA8B0DE58ADC2"
00325     "510C02C2619A542A139FA3EF7A03AD3467345D9573C107A13E7FCD43C0D51DB5EC1A09D409"
00326     "DA75462F9C71F0C9E36C2742C279C910F07CFC5CF7F98AD48D67232A2DF29A66B782095573"
00327     "57A4BC91922D4195DA9533CD3501F388AF6EE2BB3AD08BC7D53015059988F5B9BF7824D066"
00328     "DCBDC61CA588DCCF0EBDE4A96632DBA22CA0D770C61A1DD66EDA882D02C5FA284798E12296"
00329     "E89C45906D315EFDBA816FD869DF869A65DD8BA4E0B13C441EEB052EF3D0FD436E4AC68EFC"
00330     "749E0CF4C7E15599D5514E136ABD134BA638A02E9EC1FE66CC9ACBCE5082C8734196BADC21"
00331     "F4DA7621D9FA725362C411127836A26CB44CB3851D53C599B94A5E67862665D7092C43D9B4"
00332     "AD3FE20B8AFACCEDE920F440F3BF5552CFAFAD04A7D7E0A9CEA18D497282D44778FB7D5072"
00333     "832C0B77C4C51F4DCFD7AC07DC7A9863DB8A38F1C003CB852F6119BE801AD12B8BC7393B00"
00334     "640F125C734447DB2FD8B02F7F7FC7A23B84FB80F9CC08E3EF888634FFB6F51ECEE9B20A89"
00335     "941FBF2B49314DBDD67CB7A1B5BD8D629FA327AF2CBB47B5419A0A8CB807D30152FA560690"
00336     "DBAC49D6B043D5BC9D51E82C3B1CF4ED69E997050C65197F3D93E21CBE91E",  // PRN 20
00337     "D358BFC8C6AD1DC94E71D1F5D05589424275875AF8CDA2ABCC6404D6FCB7A2E0A74C680248"
00338     "27E02621C10CD5FB149FBA373AE32DFFF275CF386C3D7A04E3FE10B6F1A6F4782B4823242F"
00339     "29672E847CCE760BA005D6852A3459E7576A254B10A9A78A9F8112BEA39BA65898CFED1179"
00340     "D68211D98E6950ED06399E39433ACD898E2F6C87F5FB9D99518EF36429D447B0EF0C5B7D83"
00341     "4ACFA388578BDF60D4B1FB5A0CEE7D1D613BB9B99E36DC9636E70A543BA6BF0B3A448DBDF8"
00342     "0469494239D4B7C4979D82E80C08EF36EA67560C86665D458040CE31BA009BCDC30CCBAC50"
00343     "259E4485E570F190613CB010563F6BD24C2F1CF73F6A6844AB8350D23BBC3D1361E73DCE94"
00344     "AF83697BB817BA366C9855A754EFC2F007D99A964125682E6F5CF7FBBF687D221B5A0FD844"
00345     "477A2F87D5370F4469F76073A93AEF7812275FD4F70B2040C12A83ADE5E5D862684D119DCA"
00346     "0F75AE2B56C794968A68566291B731579A1055A84F083B3072B7BD5AC9D520F64F0829B592"
00347     "875613BDD81C11622B331289C98501B01EE1D813C0E97CF36878260F80BF88071D258B9DE0"
00348     "2F3F90B4C12BB56CBC731550B5EFDE6D97A1283EEFE61CD6E5DF312D0F0153A32DD65B143E"
00349     "C6A3F2B64E2B8FFB47EAE46BD92A6EB9ACBDD11A2D730D027A3EDEADBA5965198FD59BBC85"
00350     "74B680B96AD48586E5B17625251BF4374E28C6AB956C6818183FDC119499E",  // PRN 21
00351     "FE69433233B6067B0EACF1F47BD3AAD9783FA30F684110D1152459233896479D08A976B853"
00352     "E4B7B52A34511239961048B7C1B9009095327C86F2EA291FAC1734ED2596EF19D04528F3D8"
00353     "F2A3430A0C19DA6A70A37DB6DC034BA0053B57ACB9E7C00ED9BD6AC11339EA169D9D54E673"
00354     "9B051AF40EE79A1034D6294261E1AFFCD61B9CA5016C56B2D1172D9B2A7283E4EE0A06C814"
00355     "9E5A2DAA263A5D2429C2B1FCE75C41887DD02E056EF8724645FEC6FE7FC1EF180529B1E894"
00356     "773CF3E2E1D938EFE9CD824D914541167975F5A84746537FED5F0EBF0583C8508EA0745B498"
00357     "9954EBC4F215BE3D515687BCDD5DFDAB9814358B07038E0CB869A8C34F916FC67773191679"
00358     "C60A15A0A399E224D0B0168439386C0AEE8F5EF77185AC847A66D934CB0ED6A3467C3B386B"
00359     "A7F115877F36B49E111DE49E409468F343A98974F4EF1EEEDD282F73013EC2727518DB46C6"
00360     "751A58AE3E0D5F9D2B966D4465BC5595BC31B2712AE1E1BF9915CC0E02CA7240EBB9A045F9"
00361     "59E77DFCDADAB6248D58B47BBEF3C775DEFD629A2EED15201A21ADCA470B1AD3084924FABC"
00362     "DAB6B12FA6201E2A239AE8F1BCD7CC39FEC62587E58C84AAC15935D45261E3AFEB60016AFA"
00363     "0902DB98DCFE586513FF70EF4E3F47773635D475754A158FACC9C470921FB0186BD6EEDEFC"
00364     "BEE9C803118851F82CACBF8C0A544B0562E2E27286CEA5FBAF83AA5C1F97A",  // PRN 22
00365     "C7386F9FF39FDDBFEB223AD8B856EA2E7F3AFEDE197A61F183FF7DF2FD6DE208E71E6E1063"
00366     "FB3774B696913524F7488EFC2CA54E8B653EF5BCB7A8F4994E312DCEE99A316C2ABF3FDF85"
00367     "B8FA9BBD4366ABBD7B3D3D433C14710A95EBB3D0FCDA2D37A443D62A8361DA78ACA781CEC0"
00368     "4542D01DE7B6C6D14CDD4EA709264251D46C42AAF404094286DA5BFF8E81FA2F8C54B17282"
00369     "1054F4CED82287F29EA3D3AA798C9CF5C5A909B9FBA641A8D9E310248B0F9A1375CE4DAA98"
00370     "EB62286B4EF4DFC58B877A73D017B17AFD7F1F58D3D2CAD3B7AF2F06699B08B88FB4EB70D2"
00371     "511190158BB4928ED1735C94400980144EF9ED06E06074E2F29325C1AA316A46E8E617B3CE"
00372     "916CFCF05A389052DE120498341EE26A27A3D757AAE763046B8CBC841350292F06AFF97C97"
00373     "07CE5561F5C119E2FF6C137094F62573EB80DC13862797C3319158DDD465FBC033CAD81BFB"
00374     "BBB54D9467599D751B9980A9AE8BFC6715C5EA74859E6A10DB369D5DF83A92655A9A590822"
00375     "8B33B36F55DE563005B886EB324CEC4160F0D18938E9FE41D39234C29E13B814DDCD13CA64"
00376     "50774800924B0848735C5DE076F66EDC973FC83B13938811CD9887371470AC5DD985481185"
00377     "F1191EA8C1D3A7DC65E1E82E2318D0FF0C9AF65EA1515DDC536C5A8BD0AF481789838DA54A"
00378     "39BA56D014E12242600AC78D28ADAC3FFD3600E896445868064D1D2ACF22E",  // PRN 23
00379     "BF5202D3599D2DDAAE5F526B6B6AC469D4BA0D0BA5D79B1DB89173320F0EB68F5D9DA495AA"
00380     "0981F8022426F68519B548B19B5F8CF068A6CA1442AF77C83B7D8649DC281BF438F9576F7A"
00381     "719A902A860B9ECE9AE9C14B98859B282010A5DC90DCE612AFEFD44E0E9E7666A461AE50C2"
00382     "656BC036648B826CA9C3C7C53B30976335B097C19390716A41FD437A2098BCFA2B2975F1EA"
00383     "E5BDBB8192024C20136D2542FD89FB8F2F94C08F765109279BC4E511787496233F15F52D7C"
00384     "3BC3E98A6DC39AFA1818B9533EDE72FDAF021E2C9B7D6C74E49B849F372B1A131F4C532DBE"
00385     "3B63635E0E1334C87DDB6F3D73883D2B43E87CF19E40D6B404E581E807E6EC1A94F5261C7F"
00386     "7EFD4CF043C90A1A7E97465022ABAA1DC21588FD285E7158FD9B67EC5FE7C9E84029E961E0"
00387     "45EB5227E4726154F4F057FA337BB20DDA25D11632A7995B810764084EBDE01AF07372EA82"
00388     "FBAFE0434401FCFE05CE8FE3C20C01ACF4E9B8EAF4D50C73D5C42A95526CDC8313DBCA6ECE"
00389     "ACB457D9673565A1CC0AAE23FD6261A8943E8FB84CCEC676601A4B302A9CACDEC8998EDC84"
00390     "7A53B3CB0E12C8B4A7897D5680CB14A3D11BDBF4826C3938EBEEFA0075B6494CC714D3C0DD"
```

```
00391     "A2F5F783CF23AD2D2545C899867C1115BF4A4F559F63E68098955550BFA1EF7771598EF86A"
00392     "08C0C634B291674BB77615121BF0838DA96D6E7C53BFE6A58A382FD9721CC",  // PRN 24
00393     "BF8903A3918B3FDC06CAB4EF675F7BE3962CD7E3C6ED643386EE533C3B24A3D94D2EA2CFB8"
00394     "3F0A346FF2875DB07BA647492D47A807E7FD9717CF12BC97B3C1BE1361E598850B39D50CF7"
00395     "BE700507863BC4BBF26620FAC11D97128049BD96C5E09DC8FF3F62655D660FE66D31AB0B0F"
00396     "6D4F8420E3D2E633C571D7FE2AF1CB4E3BEE95E092B00EFD2796A3DEF376F75B7EFCBB1413"
00397     "37D81AE52939D87956C41B1E42C1CCA4317D31AB4F53DC9502A3DC774E05E1ED5008CD931D"
00398     "DDB98DFA69960A6ACD45B60895C4FBA2BDAE8BC7DB8C821697558B1E0A3111F156738409FD"
00399     "180C5A4A33B24C5EE4991B84133CE9AC089724D62DA9D9827A2A04FC103652F216A0895E78"
00400     "A960862708 14C2699F475CEFD6359428D8C505BBE8C1A96D2793802219144CA6B3EDB45592"
00401     "9B39A3E9F3AB74D685608CE3F301FE38202ADFEF529CCFF46AF36DC24956A7CD07CEBA55AA"
00402     "4C89F7913A8A4B844FD8F152C8A823CB9888E3BFEA97D7E4AAFA07125DA4F51D974A5DAFF0"
00403     "045BCE5B868177A91BD932963451EE2673A85AA8B7D493BDF25BCC2F64AEC3150D8C40C835"
00404     "AB4F5D0B7F259DF099BD6FA9F5CB198B61018B1448035CCD34E7E7A2138F437490026050BB"
00405     "E3CE2D4CF4F4F095CB97548E5731A338CB3903519D6B13A029727F047A7D00904A556C8837"
00406     "45410360FC878F77707A716D549ACD6A70A18F9EE0AA8A6EE2080608E10AC",  // PRN 25
00407     "F58CDE0EFE2356F429B0F2F9A7869A4142A6173188DD75B570F1D1ECD282E4AFBAD11370C5"
00408     "B4CCF3C98535D27D73C0111F11A84711F732441EAECAB684F2F0D7FD4FC4070749574922A9"
00409     "06E84B3350CDE5957DC388FDA23BF45F05951A393DA253EAF691940897B57ACE655E9630F0"
00410     "9856E76958D6BF7B830E0CB8182AE226F39D48036C867BEFA7E7ADBAD17C1AB45297C757DA"
00411     "4AFFBAE677B05677D60DE1D975A4F3D7EB3461B424B67B61025AAC257A69FF720CB9DAC007"
00412     "C50C69A7ACDBBCE210BAD4DC2E629A039D98E7EA037A5C344B5CAEDCDA035F28677A41D55A"
00413     "0E3E6E480CCB12B8F17062A983F4E651B4F7CB217FD06BE46747CD5418C0C81916465A4F56"
00414     "60152B3E4781DA8040D4246F9BC47366BF663CF9DA3BB247D9238873CCDC6FC62D1D8F669E"
00415     "FBA42527112FF4072262F7E65AEAC328871DDF47588A0A0DD13A4139F4145822A5917F624B"
00416     "881BFC354F37B6D59C566823F629A21C973324F7167BC39FBD2C121D2A849308D13DA1A289"
00417     "48EB59F7DE97E364223E17A30119BBC7F43E21E7DC3093F7505055ADAB4654194A77C1CCB6"
00418     "1898840125455A275A8F071273D8C13934915D379CC603657D99CE4075C1F1DCAB60B6BD62"
00419     "ABA1A10B5402A59706798002EF30ADED2F354E38CE0B57900FDAD31E7F684E53D097B4313D"
00420     "B552EA66F6D337F29594470D3DC0BC6CD361831251004DD3C5357BC0BECFE",  // PRN 26
00421     "D9086F7C272AA317C64C00AF43C924DB5DAC97F8EE3ED2296252FC4756FCE6928BB009D448"
00422     "8B9BAB757411BBA52BA6F61AF1181CC7BBA94257593FA1BD26D52AD5014C3F1A1832FC4F74"
00423     "45C8BBB77C8FD31C88F0C5D4736D49DCDFBEEF2B8301E31185793BFF87CFD9E6F7E084D343"
00424     "AB98BA3518A87A5F915BC0D76B01AF7DC1CE45F1C5280BD39D3E3D94D0A0286F8BD9FA9428"
00425     "49664E08F2BE0B93C6E3B89061193FADA0FA9485F62CA87F3E68E204186EF1187642D65116"
00426     "2E4D8E7DA049F462362D8C94539CAAD09AE4768C96ED6C2CAB8025EBB6901CBB26865E1F19"
00427     "FA1B193D47ECE390B881233578950175C85B928582D5B439EEF2F56A8C7EA09278E4774105"
00428     "1223AC182456C4FA04D025BDB33FA10C48C70EC91BC709E3CB0FA3E01DCE5FE5ECB9018130"
00429     "A8DE5D0583EDD68EA2EF227A612748B2F785A30A01014BD479DEC6256C8AD88470F79DE0E1"
00430     "432CAE448DD7049E5B7D4DF3C978F65E708CA3759AAB9D329C11FAD71204E1E92322E3EA1B"
00431     "BDD9D034E2A23ACAFA21CF490AA5E2E419197DBE990667BCF277ED61B264632F694392EF52"
00432     "F0A27C38E478257AEC8D2542938BF0713EBE60779C95A0EEC8F32A5202A849CEE8CE0F9970"
00433     "2F595AEA839531D4CFB5F5A6166B06EB64387552A1F9BC6BB97B9B99D19C3D2E1E8E9B305D"
00434     "525E7413496E40FF50CF77D4D4E2D41B1D5929848FB2F1FDDA5A39DEA0546",  // PRN 27
00435     "AE4E3B30560A50DA55AB3E59FFF512844A2700D2D763D85D5C3FD8CFEFACD4D023BD926D3E"
00436     "F2E55EB1B3831F2276EB07E5C07B44FD7D79333699BED0804B678915FE0F092DA9A62F69CB"
00437     "020DA21932F9FDF9AF332E1B400C6B7E7880508E840D62FBA07E827A23A2575AE68E15AC44"
00438     "4A1CE35DF3C3F7CA49DEF2966DF3BA89C8E90ED5E2421A6407F2EC51A3E92A3608FCBD6AD9"
00439     "FF9E5C7817E79A0C09FE9014F7AC291448263E4346CBC4BAA6EABFB59B4526B654070084F5"
00440     "2B864F9769181DC6EA91B576956397CE55CCDDBE41F94E5DC366E775C86ADB1C807B66D086"
00441     "96A2BEE45B90E8736469A371F05929D9D9FD34980DE08E00BDE2CD0EAB6AF2165D76519F8F"
00442     "2D894AC70740D2372B37407BDA4D943EDF1CBD35CCE4D81340CC97751C568731C009DF6557"
00443     "1F28B7F58106AE67279E83C3A0C130DE0C5B6C99117099548661D290C4CAF3BC60EF719E2F"
00444     "7B210FCD4381C33904AFDF96DC3A6557B42B6EE895B4D604F5F8985F454C51E32B2C874E90"
00445     "926CBC58D044D483D6D2A7C26C7AC4D190531F79993D07B2E830FEB99BFDB00AE8C008DB1B"
00446     "762F3F4A81D41295FDDA37F3056B1110D4F0CF385F9FCC7E14C34F6752A2FB17F5CD3FC4AF"
00447     "0D51E4A0AF7D28DB0D4D651156189209480054F8287266B1CB26C9E8CACAA0BE5A69C69630"
00448     "0025D160F9DA29F9EC79838941459B7B8164AAD95577A0C532EC2EDB35250",  // PRN 28
00449     "9CF0CC00B5788DD743A5F33D87E8FA5733B72EDBCD61AA4B8D0B81213DB52E7EF17AE90934"
00450     "F5EC0711ADD19E881CC330F696179C1BA464FFE6D7B04EEC383A4106BE5892C5DD1BD719AB"
00451     "3739A909A384FACA455E6AF96600AC6FF809788700DD2AB93DD228483759BD903EC002D4C1"
00452     "278808B764F018E3B740EFD821A61F5BEA2948A653041FB31F6D5D0DE0A045DA366E44112C"
00453     "820FD7FA966B2CCFD5A6816AF84DC0A3EEB8F9D2F0A912586F91D50B1AE3D930A680A8FB74"
00454     "35B6875ED2E599B87598A7C20245296C4965E2E0CF372B6ED1219BA68CB646D3E73D52665A"
00455     "AF2E3D1C4DE8D264578299B166FA0E148281C877FA9B14818759CBF7FF575307E80B739335"
00456     "99D94EAD2FB1C08A30006330BF0AC1F1C0A4EE6B07F9F3381AD7E2E469E8DA9C2D22CFC0A2"
00457     "08B58924D2F994AFC0268EFE206E0A9EB79BB51CA26FB49013B9A17017E0C08F9FFC6C319B"
00458     "B1B5AE41771443BC670EEB91D7769F9890A9B80F52CB0167EAAF850FAF2A52B74ABB1792E7"
00459     "CEFF68C0D38B01F244AC0CC0EF0731E3BDDCDAB89DF376973A7ED5D4264EE82C334671FCD3"
00460     "9ECD6E2CF869493914F332767BBE461707166A9164776D29F5EC9291F505AF291254D7319A"
00461     "A594B5F397D5BDF00BB840C4DDCB425F4325ED8AB77E57BECA3441B89414616671692EA88A"
00462     "89D2690A4B5FE958F990BD84A3884A60FADD5DA57EDF01865F85829195460",  // PRN 29
00463     "B85B6E754CC8F6805A8A19DA104418D9C134C8B0DBCFD5DAAF5A71BC047A73BEDBC192A453"
00464     "674BC624959BB76E44C5B34244D4736ED3F0F3C9658FEC0DA5437E01E128795EDD7593D636"
00465     "CD73FC1780B37A381502633CCF2EFDA0BBB494C1D0FC7F602DF8C282F55E3828E81A92458E"
00466     "B16B74835040D8A9C8F2DDF180A617B0592344B4373E1B526C9706B843B0CED4D25D7324C6"
00467     "FDD0F33133C00443638E6249061C56A116CEC7822F4512AFAEE52CE8F94D8547F72612EA8C"
00468     "7D160C65FA3BCC92BE01493706EC4E5F203F0BF85C52F417BAF8AF490E50133505685CE63A"
00469     "C5B173E07D8DABB2D439C6DC18B41B9CF37D02C92AB5C2F27EC83AB6B2DDCB7ABCEA30A95B"
00470     "BC39E9FD0CBB28118823F7D0342F1EB7B45FA6BB3A50223D0D7B14E975E7658352BC9288B4"
00471     "8AF1346955F4551F2ECA47D423EFC63D20681057E5EF234D061A5E6E234ED01F3DF223A0E8"
00472     "B4DEDDC552C7DC3ECF663D5011FC907EB4A7CF746AB9E07C2929B7427DFE9E00B0A1308881"
00473     "912635A72EA99927F343EBAD32436A9B8EB1934AC29E79BB80AB3ED9F5CE39D1E43C251564"
00474     "654365DA43FB8A0FBA27F2328D82445A1EAAED67B92716147E859064AC326A42DC7880DE82"
00475     "FA782AFFF9C59FBDCE088746F8CEDBA288BC8C2C4B458782CC9BE63A86168B671BE99A09F2"
00476     "217B7BB2A7BC88651C1BCE8A0B89316ABFE72B22722273AF570974D8EDEE4",  // PRN 30
00477     "DD40DD438251E401FC926CC6968393415D52D521A5BB34D4272D6BC7B5431062B35112CA70"
```

```
00478     "9C0680CBB18EEE053AAD62B2391C9E9D580562541A453ED936CE8E88DFA61A88CA3BEE66CF"
00479     "FF801785CCE863ED9C36A04D2DC8742A81CA55127B44314AB4E687ED921B4881CB363AFB3C"
00480     "CE7EB774E3205D4591939ED7D3C0C508A31786421F49669E120F01D35D467B40F85F2454F1"
00481     "3F591F3B830937421B5C8A6C20EA878971AEC941FD99CEA92FEE00E5DC2264987DBC549EFF"
00482     "3E4A26AF0CAD7421C4256D107A3E8908F67450960E4E41FD7E2E84F754BAC81C8F5F1D6F65"
00483     "0DEB3E6EFF6059836643209E3880D7BDA701869208D8E4BC8D0614066414DB3F93D6EA1879"
00484     "50285F55BB7A1B026EA4BFCAB4671B07704828D5CBF9730EFC99E68E91F1FE9664DFA73297"
00485     "F2D6BD9497DE04982C9FF3730BB6FC3EA2053B3F45DC7FB587BA19B3C6B7E780EA5F25B45B"
00486     "B727174D4CD3B401FE1906360BF0B15DB13B62752F82EC62226AABC83C1C26376F8366BB84"
00487     "9DDB65958AD969B25654DEF18415189930333AF47EABEE3CAAA936F19E28A205F3CDDB5CAC6"
00488     "49DB6A90483ACB63A24EA46D397508EEB5DA94E9C883EB0451D036E28CC303D52B1BB31FFF"
00489     "582605F340D449508959ED1FE2FF0BD22FDF77F9680D6B5647D59E7E6A003AF0C6A95092F0"
00490     "DE43D1252EA6DE00F288BCCE3ED9CE273DCB4F3BA7E8D17353B8ECA24F03A",  // PRN 31
00491     "FE38B1ACA366B4C15F3FDD4DF0E0274FBEFDA0042BB203A4F6627ED9E29F405379B2F2DDC0"
00492     "F3B02A0CA70A9499F3CE82B87603FAA347B7052CB5D13D9DE84C114EF3B8F62418FB1F3E37"
00493     "4B997127667FD6BCA2E2F9DBC04ECA9D908CD37C62F08EEA6F44B3FDC149465AA8037D65A6"
00494     "C8B9B8B3D5E9A40578E5EA3AE1209BA49E5E2AC615C59A2D71AC1605B98E39A5E66A890754"
00495     "C7D1C07E06DE78632587BADAF7FAAB0A529AB791095DB0A708B691E9D81F2CEA8F07B05495"
00496     "528B9FD56F77A4C8209DB972FAADD9791BA59F47C06F241F50C0619FC04F8456339E0AF331"
00497     "310FA4DCCBEA0E5DC2795CA6B3ADD0174AE4B30AC0428320ACEAFF68F73ED11DC1BC9F0237"
00498     "BDC75F7F48BE518EB3305CF2BB898B329716FC9ECF7E99B510B3309808735FD0A77B15731C"
00499     "233998F9ECEF46E2CAA6E6EDC8D05B943ABD17027A80D636E535038FAE44D60AAEC5406A37"
00500     "2D62479192FA84D844520C6774CC589FEE16A3A5549495D968AABAABFE4DB94F5AE0C54E60"
00501     "3D6DA5C3056769A064890533EA8EA1E5D1CD410CC8DD4B1D7E0F5F787232439AA4B3911C5D"
00502     "C792ECB873E8105A1AA61C627BE57E809C6863073E1E19AD8B987DE97D88A817FB43ADBB77"
00503     "51E36D1F0E7B70B3759D6EA8F2350D10AF38C331E22703B2B5103C908E1D35A8E814E45BAE"
00504     "81DCA0530FC3525CD640548245C259738E749E195B006081A18C45475F906",  // PRN 32
00505     "961408BD52EA1C6A9F340D9109B2388CC358BBA2D35BB6AB672A9C16F3AF968BE4613BA6B1"
00506     "3D115B896BE71CFEC4A4AFC0BF5D2BB1B5DC19405F40E60FDF361E6CC362CDC28B75B8C30B"
00507     "468D3BBB77F3FC62869FBB635A3F7AEA63420CF1B80A4B3813B2240B83ECE999808E1394DA"
00508     "2881DE2DE62C870EC163AD8D81495DBE2C7383B78E19AC506AED9F3BD1280A77F2D9C55600"
00509     "BFA258E76761391145D45F74253C6E14BAE16179884F0F0EEF8150A445BE1B5AA4FEDC2198"
00510     "CC39763A3DB473C1CC4263CE2DD587447DDA6BD0A496E8DF60859CFCDAFED2EEC5B1E77E68"
00511     "F449ECF129AA17395BA39392EB610DF45134571BBCBB26162C83FFE77D188160EBCF598EF6"
00512     "F092881612A04BA9F4215D429BB521E737F6C3048B5D95B20AC37F79AA99A12CD046926822"
00513     "8463C317A1F31E63E4754890F20B4516D179342A76201402BC0DB2AD091A70B4232473343F"
00514     "E9E0002E59044C5F3B6E3D7368DB7E7F9F42E7A4A1942B1161552C5C84390197A54079F570"
00515     "B57E8EA8A30659FF5A61048142E4368D5542B968966E54DBF91D3970B9A3A278B951D6F914"
00516     "246FA5CCCC20DE53713D4830AA86D758ADA7A8747FCD455B2C320FF9E5E7FE1CCCFD6F7928"
00517     "884FC0B35F34118B4E7E6F6A5D6E3FB4E3DF90CEEBCEEF9D7D595A4C456C373C5356EF0DB0"
00518     "E6F8D61E413F80E4C32532BAA39170AAE2FC606B7206C379C4155031AF918",  // PRN 33
00519     "E03A069F63220109A77232F2BAF29D345737AAFED2E5E8D5C846B937277DC88392DB2D5507"
00520     "3CE063F5FFC5717BB89B481B8C86EC01808A85ADD78517CE12DF776CD3F02D948BA7952155"
00521     "99CFCCD9B4FE56DD681A59C71D24946225171DC18E47BFA9068302BA428929790EE62306C5"
00522     "FC8C10E71F6F372105C9421A563A4DA704E01FFD71E46B28C6EE1B7CDF95BCA6A794E8CFCB"
00523     "6BCA6AFBB67DED5C1267A12A4752176C3C1E6C2F665194C7F11C1CE6CC8481A5A966719B57"
00524     "124CA33D8EC9862AE5485788106562A0BEBD3980AFC4469BB1FD1653678192B3022E26CE87"
00525     "88C68F202D506DF098D49EBAFF7741CA96A02BD68BFC4FDBE24D34770FBBF8EAD9647C911B"
00526     "2E6AA705F0236301162CD2B41B88BDCFC2A79EC546982761 26D6EC0213931609985224BF51"
00527     "5AD3221FB5E8A1C4CECD51DB4AFF612157ABA6640866DC6D3602C446B8C6EAEF359BC8703D"
00528     "5EF9FE8EC7F5F2685C53924C6F1F71519E0AB589EEFB393A12C06B906402FE0A2FBD793272"
00529     "BC101D3B57F10A52C172E505B74F2A00BD5D2F7C7EA2883908434C125969A62F1B09F42E57"
00530     "8BE22C08E88B11767D97C4C0CBD2BBD98508AFD591EB0B0C846A95FB72159E561F9D87DB44"
00531     "6E9A19AF378B6DB4DCCB49FF8DB547B3040FFD0171B87245CF73A0747287B6FA1F1E4AE995"
00532     "18E8C53DBEEA942AFEF75B69E4790D75CC8A5181C609BEFAB641E28C07082",  // PRN 34
00533     "D8D7DD9242E54C6DCDB2A717A6F33226A94D5794FA0BDC401F4ED842A5CFA4AAB462F70323"
00534     "9F684DB9B95E5101A5DC6067C7062AED9259CDB2067BC815C157A7E150F8557C3A54DDDAE9"
00535     "4E5C569A1E09C383A062B601F920EF4D957F4BCA8E329123BD6FDC3B731361864CC139EBE3"
00536     "C68FED0F40FB127D9D1DC071DFA1552ABC6703494A632AA1314984A2D4B7A8BB32C555B79D"
00537     "EE013A66745AE15E8E5E4E129E44A119203425F2ABF1F03CE9CB33C0BDC3285ADFAAF4D7B5"
00538     "1EC31F02D1E654BB10F0CD97EC3E389CAA34398166B4D5C9FCBC6E957FFEC9CEE4071F90EE"
00539     "2926FACC8FDE884CC6FF1F6EF1D3CC681FB44E45C5CF681AAD13226DFE19E22CE81265F308"
00540     "8D193EDB098988B2640EBD8D9D66708E1E9880DC41A72FD3D64792B14AA18C13E371CA17FB"
00541     "46B21DA9C59FFA2FD075852D42E2A578744792EC02F9A35F869912BBB44DCC648CE6075DBD"
00542     "F457A9AA891488A5450FC719778739AF323E87F9633E621B404F70614F77C65697E71F2816"
00543     "75C843C523740AB66756E4DA784F555B5B4D797A06A0AFD35A69DEEA6E948B9B3C2A62D7B1"
00544     "D56832D9CCF56F2680A5A0A3037F4E0252413FB86520F2815C8824975634C0889A486963A2"
00545     "C874638559E14F780A7F3E2318B88B2C2010174150AFD4BECE2C5FF2D37BC2FF791F4A3136"
00546     "E19C373FCF03E471DFCCED19471182A93E91ED3EA68C402234AB2B00ECF62",  // PRN 35
00547     "87F0D84B811E4635AE8023B74306DE789ADDF1CABAF5F47885CF7F9A33F2C533093A339EB0"
00548     "D5E05C0763A215459CFF0D31CA92573EF2074CE2B6A11FDD6BAA3C6FA100D23A9AA413BAAF"
00549     "DA22F746CF74562F9DB0EF2F7CDF266142F177681CF2EF388E2E9AA012459ECCA332B779E4"
00550     "8CA44E536082D59C3951ECA42F66B600D2621BF5F3584B59DC0DB76EC66ED7D00E943BFDE7"
00551     "03E7D5050A8F263366948253B3576311CE88B463791DFA6F401ABDCFD7FA44B158AA97EF1C"
00552     "FD7A8802662A633EED958052DEDB12A6FB353BF2224ECC2322937DE3D39DCF82650B18FC0B"
00553     "C2BCB8CC456141C9F3FB09A0906AC1EB77E50E8D260041E4B3FF4BF3A53F7EB62E0FC503E8"
00554     "E38F185AF795F67FCE17665AEE29BD0D5521024A70B61446CA4CD2B3DC274FBE72A9AE29EF"
00555     "67A2217D6CC81BD8831B5160E4E81238B379B657FFB49023B040B6D504B287F684A0A0C073"
00556     "04E6BFA21E8D0A7629BA32F3F2BC9A33DEE2EAC1A2C22462EA0D24162543945B78FB6E26B8"
00557     "6E12621588735B32A4F9A50157F20BDB7A4B6D151B3F28B40A03CFEB3CCA635261ADA2295B"
00558     "74947F1B1D676F6C014AB362C1F5AEBB439DB137D034D00591235B6F8D11C735A0C1964B29"
00559     "D3002D5243374628FB488A04EF245E64F598EB6ACBDE8B87F2FC5D1ACD105460C26BC6E1C7"
00560     "DCEC2E92E33F722E0A613A86356343EB111D0B166AFEC5C7A44B81A607A24",  // PRN 36
00561     "DDEC47E0FE3E2F4205206CD673EC66D7435E8BD4A523A8681ED77F51453B904E4468C70C22"
00562     "24C5F1D01A1C5ACA89BCD72673F82955FACFDDC4DA499EFD8D18A2BCC5035B0C69D095AE0E"
00563     "F1AFDC389B6253BD6FC83D3C37E809EE732A87D065207684332CBB9BB0519CFF51B553B368"
00564     "9246EC1EA42E236773A5AC4A7BCD37381615F78A41E5B181AF5C502E22FE79436D6EEC07F7"
```

```
00565     "FFAA739356CD9544C7DDBC42C56BB1C9654419039627039 86C93C6F8135A8EC42A89DCB46E"
00566     "754EF5B5250E1171398FD31793594831B0775F2A39E63FAD99929A0F257ADF332C078B0B7E"
00567     "209229ADA46D69512FF2DE930F962B8F81EEFCFDD358FEF8E4DA0D49F39C43AE9D99D8052F"
00568     "CD60305FF4D9FD4CAAA3FA6BF258B867E3F266F1BCC77390D02132E370AC79B34FB37F12D1"
00569     "02985950FA5FD53D3D4783DA5B284494EBFAB51DE6CE259E27712A0EB4D78EFD6573A03D62"
00570     "9F29166B902795FC8DF59050C9FA48584F0EB69293C7FACEA8972098E1367D183C0B91C355"
00571     "0852AEE6721A84341612E820CA4F53C71A51774134923FA00FAFD84811C07620917C1966F6"
00572     "A26A032B125D76BFA149FD66EBD18F7536285CF94A8A750680D6BE0F4A4E7F2433018796A3"
00573     "58CE465E7507F186752EA2FFB01941902129936F7B18321902780E2C6A5EA7D3607227D631"
00574     "5C570815BD0E808EAE242B554824ABAF30614CC72B74E647D5B5A6306632E",  // PRN 37
00575     "DB29CD5101A3FB7A651BAB94E0CEF2BF737BBEBF0F755AA42F1C0B2E6A2E00422458CD8E22"
00576     "44576F139E811B2469E1B8F10FD95A988FCF872607A9B5CF81EB5875C427515D8D6D4FC58F"
00577     "3C69A92AB4932ED3D1CE6191E648276AD746F12A0D7B1ACE96B80CD2C3B99BD1094ED29F70"
00578     "20121076A6E6D5D750C39633B00F3DCB4793A27B4838C492EFCFDF94D955ED33814C954FFE"
00579     "8B8D29A81C3C62BB2E95ECB393F06EBAB479494628027CD02E59F1AA32F78C53142137965E"
00580     "662DF4B3EEB0B92A6FEBDC91F8B31D41E2EF69ACDE1CFF2AB49E2A4B12C275A20439E30C69"
00581     "0D14D4F661C81308F11EFA3D014009D80D794D9F2C8B51E2E6C83686DBB07D1790F56D7C8E"
00582     "E495657BDC081A63B1353F4C4CA74CC0D02CFD7BE60E063A33D1A4D9050A9100F0E1817047"
00583     "99357532DE152CA54FFC089E8DF80F29DFA14C19C9D6B7855482622504F95A8948A5DD640E"
00584     "E88B87CA98F8248DF148AE4F992BC7D1FEB3BE07E70F22CC052D92FB263337528C41ABA642"
00585     "C6B1C6F8883B21ACDC69196733FFB684451D188107C1DD219AB8B0AE5F49628A1CE32FDBA4"
00586     "B8CF02F38513E37542D020F4BAFE03EAFD7DCCAAEE9E28258322E43DD47DA3AD625DC2795A"
00587     "DBBBB9FE0619175739366E31DE6355023F8D74500D2D032B7A53630F37FFA80567BA36F771C"
00588     "4ACD71D76DE528FF47E8281ABCD325E4C4620D0B73B2BE2787A7F6F485492",  // PRN 38
00589     "E8F24D8DFB54C5BF909CA2CC5496AEA76E583D2D865259C356E64E76793290BB00E7029A14"
00590     "6E1E90DC0C45356E13EF59D60F20F080A0FEA743FC1C222AAEFA3E55876F2C9E6F7BD29C09"
00591     "AE32E80B15DBA0E6B594E951EAA3BFD166DFCC17AFD2621EAE6539C74FD776BB998C24C30E"
00592     "DB3A8B6814DE088E7E6B7CE9A64EEE9ED8C9D987A21A3BF35A9D59171DD456D8D0D7D1BEB3"
00593     "F37D4A31812BE00077F0F0064DA56EF9C4D36D1D3CAB4D1D4C024665BBE1DC227DD29E1CF8"
00594     "14EA65B64DE60ECA4AB9B19F937DF15914F3CA577EB3A6A0A2C8ADDA53DE536721FFFD519F"
00595     "C5E7CD9A3BF4F030B1EB638D6A0F8C4E24085012D758414211585EF6E0DB8C670064885B67"
00596     "256BB6AD7E12E380C5E25BF58EB1731E935899C1FD2B20008CF87529AA3E714BFFA86D8B66"
00597     "AA94811A43ABD868A7711C4FFB339C01D72D4974ED53E7DDACCD36B5F459DDC05E9D475D3E"
00598     "2AE383F6FFC9C2A0D1791B4675DA843303C96A98ECA88B54735293C4E1906AF30221EE71AB"
00599     "58F3E38271608DA6AC0A488C0850DD86F8B6588C91589632EB781637C14D9D24B57ABFC301"
00600     "790B3C11C1B2938845F08F7280388E0B9D5A9682A6FD40374542634590F336E42D8FDD92F3"
00601     "F6C82BE4D3B953CCDAC984F6C94D8F2FF70BDFBA63C922060C322FE34188FF70A37648C362"
00602     "BB68B06D74A2012050FC007F276A54882A8A04DCE014655AE43E448639F56",  // PRN 39
00603     "FE35174BF6B56B67D39271D92AE0DC2B64FE31AAE8D1135BD8FD308D7E281F3ECE84784423"
00604     "A48C1362B5852719023F8861E861AD8B22219259E357EB9CD8505A66EF7CCEB53636B47D38"
00605     "A2AF5506E1FD72D3E30A29EF897C5C44A271EE562D67B279806E8A5DCD78DE538D8121CAF4"
00606     "C217F8A969AEA50D6FAC68066277242B1979F1A6B3051CE5B9949A11719F556EBEF844C808"
00607     "E5C1AFCE5E312C53AA9DDBFAF7280A7FB8502D2C7D1B91614157AE0C6C962F868D05D04631"
00608     "31DC841169946B732F8000E686467BD5D8086CC249693FF9794374266BD6A5C8AFEC65C66A"
00609     "834012365D60179450C58FAFE724B8B9E99C33900A86649B04CD54351D6C4CC7068B28F070"
00610     "417CC9C4430390493BEA50799FCBCD7A13BF75947C597B3D7AD486E3AF99CDD743B6230BC4"
00611     "73DA35E6D05FDD17F7CB8D04A2B00EFBAA30946EDA96BED467A45EBA54578D9001637702F1"
00612     "DBAFAD16D2608C475B8DC7994DAB93FB34DF2237E4A13D0C04A6CACD42FA9463674AF8ABA9"
00613     "7CF511A82E8E61F8330004E165D753323F4AFFE598E4D108DD8EA8EFA45693A2F9EC8335C7"
00614     "56393585E052FE5D150A58F058CF8C3C720F37112DCBC6324877E87541F06C968C46FF846C"
00615     "B512EFE65CC401174BBD1C977694ED7BD1702EBD2D1EF23BB56D0D0C1EC2D90A27CC63F0A0"
00616     "9D83F8B0A5ECC181D5D4265911BEE77EB1DCB4780B53CE74DC7A42252F3FE",  // PRN 40
00617     "8FC54C96AC76BECE7284BB32C63FCCABED194E82FBBCFBDFDC6AD09AF95DCE6A5D6E13CC06"
00618     "B247E38253D359A6C0EEE00A660BC3BE6FB217B9B554C90186711B1E85117DF0CA17463E13"
00619     "2333B8469ADF72C3BC6F218A96697E172CE2E6D6F4E70EEC2A0CB48F0A2E7991B1B90A8508"
00620     "8CD2C59E3C9AE7FD939592C14AFE9A13829DD97E345D1AF92AB46BF196DF906BBCAE16F5A5"
00621     "8A4D99C7D586869C81E182352C210E3B1EE822F4A95DD0BB3B285632978B18FC1CC29BD58C"
00622     "2152FD3BE73482DE1A6A79C3929DDA1D20B6E99E25A666AB20CD371A85DD20BF0A76C5A810"
00623     "41EC943A89A94AA64C5207B3166441FFFBDF4AA28A2A88173F280F3A2838A98667775F597D"
00624     "0368DC0BC53C1AB8B7670EE23E0ECA08CB09FCA68FE2153F01FCB7061CA1B6AEDCD815D611"
00625     "FF71868F50855054085AA1B7CA3309581532D658C7D2D6069C3B44E0E3CDE45222DD9EB40C"
00626     "369762CDDAC6D9A6923FE0EA7DFA1DF73BFF8B60EFDDB8863538A38528803F3EC27E09C87A"
00627     "2D0E160C2480F7D2003DA331FD5C7FE05B582DC9AFB114D2AD2F822922067F1FDB3261078F"
00628     "33879C4970350031711 65FB139F79BC508AB9D3250D1B53508410A01C35B53179076D9F46C"
00629     "5BE1A26DEAA2F9F71F442FD7D19D34979F6ADF96BECF1124551D4806E7136551291352748D"
00630     "D2D58443978C3DEA5ED0544E6596769A760B476B9CE7BB09543991EADFAD8",  // PRN 41
00631     "BA63AF12FA5F7D03F714CA9AFBF19375D2A7E31EEF4A9E10C7C2F8650552A6CD22FDE0E012"
00632     "D46CEBC773C87729C7E746FA4C0361099D4876701C0B2A16EA2B5A6B750CDCEB573DC711F9"
00633     "5CEF06B8DF516CA2F9BE6387550F22502E53A772436F324569B25BBACCC781D30DE25B806D"
00634     "369AD1EB1D1137EFACD0BE765DF4D06E177D3E3F13E9A3165F269244FE8B130E9066B23474"
00635     "418A5191ED481BC8974074336E71BEB02B1BC34CC620BDBC24CDA8BCEBE068416E5F5B5A26"
00636     "3A51CB3F6589F77D4E607939F7A0FA637858B96189A014B5688A9DA32FE2ABFC31FB00401D"
00637     "E475BA07265B3FD1890ED0FDA487D7C616E099B4EF4E47C9F9114E6BF8CC757C92C02C4654"
00638     "6130ECF7CD44160F55A72831692A5E69146086C43A5F043BD1184965E2A6B154BD7B8BFB3B"
00639     "4B28C081B0F349900492C703913885428A82A8D2EF1240414CC0B6D56D969BA1A121D9D615"
00640     "84C6A880D533AB58B94C85F07732B5EBEE7FC87FF3AA279ED703858957A1C1501D71DC5420"
00641     "C24BB2D570F7589F0AC5B6EAA87AF68442FBB38EF693D2200E73DC73F58CFCBBE43FFCBF76"
00642     "DB4241FC4A7B131F29C8BC0F77C95843D2FD2DE39E3D4D44237DB39994380C5F1A1A3AFD92"
00643     "7F6B736D585112920296B90CE31D58BCAAF8CCBCBF15BB36199B48F00F308F7E8264D039D6"
00644     "DAA6E848CECDD1544323C9654232F45689076B7C4B1123AE0FF152124E930",  // PRN 42
00645     "F2CE3D09E701F60716A413AF06E14E6FB7BBC29DCCD8273A083B00D429B57EBE9FD5270EC0"
00646     "C299D243A9FF0FF73DE19977EC16C2772C6B7FF88F83481924C921F49EEE41121DC79B7C23"
00647     "295EFA50292B11C7D45D66BCFADC3C3A390E03E13CCCD51309BD4061782273CE5D0E1BD3E1"
00648     "1271A3C52654D2B846B5600D3E68D2C234807BBC32F1A350839500DEB6D387AECBFF534460"
00649     "6CD04BE3614262F068FBD256B2855EABD5A7C0AB5C28D5719C844F2A6FFC500628CDBCE869"
00650     "D2F090E588B57B796029A84E2319FB6E59960553B62BD30DE99AA7ACF508168EEED8DAB999"
00651     "6041BDD78BAA2A1CAAD2EF366F4A3244DCAA3F06EC9A24572CC80C3CF44B557CDCF6279165"
```

```
00652     "414066D6B700DAEA8D361034D9DE455A55E55CBE39898B65651FE709506D1A1FF67585D0D3"
00653     "E1B2C5EB599704B3925CEFDA45C33D92C9ED9F0C45B7A80706E6629594F66A0F74A4A76749"
00654     "3B73B23AF323D519B0D05EEC62C6123A0829CF612DEDFB7F275A55F1629DC64D2F77125A1B"
00655     "DFD7B9C213C51DD9FFD83DEF42E87AD4E0F9E5ABBAA6B120E131F9E5A3097F7EC766539C73"
00656     "3522FEB0F02604613CA1190024CC1C3297E7CDC3C514F8C3787943EE8CE457B516677437B2"
00657     "212A19629A0A04B0DAB8258A17B28F52BCE915A68015222530430 1FD2BB1BA2A557B3E3CFE"
00658     "6553CABBF79A47FEC31DA590308D156C3537B97116224F3E21EA3841A505E",  // PRN 43
00659     "AB2F844287488AB6B0F47218F2A4C54E6BE79A80F1209CD747FD88A575EBB11F2F0756E2C2"
00660     "63E753D9388876E159EF3BBF99448D0865173572F4B6A03FB72244334E4E861E0495AAEF15"
00661     "276379F8DD88AF9313096805D5596446B48EF1F1BF2ABEDDABCB1FFC98ED408846D85732F8"
00662     "07196C9CC9B283EA4E0D78681C73CB6AF89E5C361476F84E979814F30C0AB03E97156015A4"
00663     "93E091EC5D854E8B08328581D80091564C25D6A714407D6B591F17D3769 53F18308B8BC12E"
00664     "AB5B5C9FC11AFEF293109785CCE29C978955CAE2601A1DBAA274B72CC18CF27FE077A01232"
00665     "58A1B879E1ABF84458BB10652CF7310278C7FA11BDC5D00DB65E5D6A13D77FBF9D02DD0574"
00666     "DC7AE3849C7C47820126FC99B9766920516AD1D63E7EAA366005714D6C1695E731E88D43A3"
00667     "8B615407A99B32A101C4D417D5E36B886C26EA1CBA9FAB0C0404687818971454891 21EE47B"
00668     "B2FBD6A064CB325AF5CECE5D5A4BD590C70C7A9B068D318247D8FE16A3609C94A8D431E987"
00669     "2E26A3CDB19DD971971AE1FBEB2FB76F7A71AFC5815CF7871BB5EFE3BD0C49BAC441ACD9CE"
00670     "1DA4AFBDB972659D13775AE77F843259EED57A62CEA93FC449EEE1EFFD9AA2D36D8FDBC06A"
00671     "95A6AC9B067468F1D813F2DB95D3456063D5B23395824385D5C68D3F869C4F639962D599AE"
00672     "B26FE2D5148A51579EF385D31E744450E11307F0C803C70ECCA93F831EBFC",  // PRN 44
00673     "9B1066F98BA2299266D8C1351E6945D7EAA658BCEA46D5AB353C2CE48D01E915E740DC90E9"
00674     "CC487126AF9FA722A335FA1A8D11C2719F07CCA23B1DC5E5B9AB198077177BAE0B5F35C146"
00675     "66FF32926AE0239F21811921C77FFB56F7B218A353F8EA67448F61ECA318196B1186514D27"
00676     "416EE2F61764940A70300B69EBD02719191ACAD969B1075EDD09EAE26F883AC99DED1AE935"
00677     "5C2F30CC656934C0175E824E76947F83D791CE7A2960FEF547CD3BA94CA064659D1F68C0DA"
00678     "8C13AFACC07F3D5A653F259DD141E2ECE9B6D25F67CD38AF3F802CCE332049DCDAE1504061"
00679     "C2437C7D3BF4E15D88185D4114E96BFF7EC673A7AF8AFD53C979C89FEFACDDF873686D892D"
00680     "EF5FB67877734CD981C16684320AF392A464C7216FC5C8B22BC29C8430151FACC94AA1513A"
00681     "08E1FF2F6A965F68E368245510B975595EFDA4E80B7FAE432FCE9737962974905F367FF637"
00682     "121237B2A404C42A5A9318DC5ED9CFBBC2EA9C17A6CA37BBD98F8771CFA7EF58B1EEE40BA2"
00683     "D6C2031EBDE4AF0590AAE8FD10B2BC02A254C97726521E4D4D4D9FFB74FA5D06F8AA0BD436"
00684     "31FAAFFCBF01FCA87C9360C5E6A62A4B025B894E2C30F9003D29B642335DB0A3F2A3436A91"
00685     "7B61776F96292A06A6B4DACE4F6F0EE0232E1C206427A0386075BF50D6074006D45E3A3CAB"
00686     "085431032AC20D2690F0435D257D3E2C93FAE49D75FF0F320CB4D348C6288",  // PRN 45
00687     "8A7F8C99EA79E1B2118E3B203F72FD8921BAA5C336B006ED66AC7181575D66ADE21D00FC8C"
00688     "7DCCA7DC9E430D1D086F2922487B5B025AD38750B4F60993368D829CC361B600ABA990E570"
00689     "370AFFF9BF171610F2A0B0B93A0A3AC54244073A0E816DD691BF1B0041BDC165125A14C621"
00690     "E01C8F069C3E05F48FB77E66A998C27A87FEAF07B5912B303A98AE5796F1B5D4E2EF52F38E"
00691     "68F0EC5786C19ED93612D7152BCF0CA1A3044898F9347FCA8370CE6E4A0510750DF6A42C1C"
00692     "35FDBD91BD2A26A60FB229CC35FC45525D12A092505D901A4F9E1E8D42D25C41B8FDBA13AF"
00693     "ED8BC5566D4F3BA13A779D7BAAC1E25B6710D7718B73A641F23E1D22CBDED3B7BC3AF7D92B"
00694     "2DA1CF874D908CD8590C80F1D5055CB8228AD964DE099A4D037202C65D197A35D8A268D8A3"
00695     "BFFFAB39B93615DA295A09AB979AF925D895CB60B5DC5580055BC4347F0DDCB1090DAEF46C"
00696     "8ADD1009A5126362B0B4F1FFDDAB4A00AD8290EB84F76AA345DD73FFF7733035026E3CC1D9"
00697     "9428CA68ADEB8CBE98E4A630F99F4F33E8AB66895AE7435D2E84EAC95CF19E9B440373EC0B"
00698     "4D4B2CFAD672C7FE1FE8CE5F34F55B016F8B115FAD6CC7B53DA7555E67672FBF6BDE324AA0"
00699     "9FA18F13D9FF1041A12B04F30304774B07659F397554E66CA589D9D9F1F262CF659F718CA7"
00700     "F443B8331BDEA8C3980045562D909EA44E917FF5D47812A0390139B2A87D0",  // PRN 46
00701     "80B35D641CF6EEF705D51DAACE1EA5EE92057FD497B0D937C7CE9A546BB32DE580F8E434D6"
00702     "44F191798A518785BFB9471AAA5D03700CB0B7B2635A6A14750F03DA4FCCB1B363C254A529"
00703     "4DD8E7943E4CDDA318C4B92B57B14EB0F8EB250686CE129BBB2B18EBE7FF53C9DA9C0B664C"
00704     "527A3C69D905CC6FAAE8BEBA7D83294C1656DA4B8308968EE49DDDC2D0C71A17B02053027D"
00705     "7DB8F4E77E3AD1C80FFDBD37938876B671D80E99F5F1C7BAEF50B7E05CC0CAB8979A3A2A85"
00706     "2A7018673292CDBCECAABC74B839FD3C084682357A5414E431C1F25E34850FBC779285854F"
00707     "C6AFDCFB7B7749E0DFFF5F93A8AC146C873B407F2CD6CCB461312AE35DEE8D6D51347B0824"
00708     "156DDD60762807A5C132C0667FBBCA7489058C47A156B2A50CA5C24B894C1EE7C44B871791"
00709     "76905B7657A8E95AF7F2EA6C6D2A12384CAC9E6D14253DC5C31BB8FDD2462581C109D2DAF7"
00710     "2238E4B63F436DFD7DD5571548D2206BCB8B837D8CE8C9C3B3066E46A1655E3D84AF42DE10"
00711     "51DDDEF438821E0F0C1EDBFE148356D707036B269C19C4CEF4C4BB4048364E2A3886E38B42"
00712     "EFA15F22CC8F92D802EAF3FFAF9BF45247DB76C03E99F662884DC2A29EE488023BF0EBD465"
00713     "39DAC307410846B25280349106CBAADFD658A066C3664E35C4C696726140275AF7611D2369"
00714     "ADB8258EBD2279DD24DBFE002377B9A3C1B120890A6FDF20597417F88F15C",  // PRN 47
00715     "ED6621A6BAAF64578BD988D08FA8D3D2873C87C38AD7EB38C0922FF3F7E59F0A8DE0074469"
00716     "8939D42D459E6105ADD3C77E9FC4911F572B3856C9AF6C15ACE704F026C6B2FC8BDCD9D840"
00717     "22DC10C47E11E934BEB7E1DCE8B6A2BAECD384D810BC3AE587E7EDE57CDEA908DDB0208856"
00718     "24BE042DE0ACCDE1511CE38AFB6C9DD812FEC339B8137D88108F07035A3869CDCFCB9402CF"
00719     "96B9E331297B644DB13DFA88F60605E067B9F35607D2D75573E0913F8080EF603AC4B7133D"
00720     "836B84F32F48534237CC559ACABA53A96885D297CBB572BAB1C3275F7CC7A3AAD10F29E727"
00721     "B28BC29B038F4003F8C93FEDC88C63D72609A5330F36F4986CD9536812676A89305C6CE58D"
00722     "4C49C088B5E273F2AF77A8D31D1B8574B9DF02CAD4930D7A7F76C067E7160ECD1A845FC6A3"
00723     "B508708C279A2C94F29108FCE34A63E5D5554FB530522394BCA00508DDAE6039ED9F476097"
00724     "81CC60F6E211BDF29E6F44ECC873DD24F1D37E449D7BB7D6279FA7B9D08B5FA8F9364EF6D4C"
00725     "8D129A5A97A8BBF17DBD5E64C4A31426881687BB79285B09290DE61AE40C295CB1BF3C8AF7"
00726     "56E88068A777ACCE3C4B6E78C62AC1E8EAEBEED0AF3153983214D7459AA8E254633B52E5C0"
00727     "ABF4647B906AC50A62543710EE92B335A7162B540AC70F2B2F49FEA892BD72DBFC5F7A35C3"
00728     "AE56636AF2887BE680FA63768C27040E888202F700007DE648482F5307C56",  // PRN 48
00729     "914BDB196CD56E3B7D7D3F1D7A5E4B0A1389578F111449DC2DF643E6E29F688227C3C07033"
00730     "C2A3818342B229F63C229FAC11EE1AB6F0FCE8608E03B46DC983318DF15FD8DBF2970EB342"
00731     "BE2E534BB0455BE58290A48FC60973553E94C4CB53566CE0250D9FCF055936523A8ABFC928"
00732     "7DB9DDEC54710859DF62829D2B6A100358EB64E6219451868D6BBC2AE4DCEA0C0E338B26B7"
00733     "48D4A1A34AC16233046CB7D346D0D79A3CCDD4CDCB435B9B3075AEBEDB4C0F18C5DC006F5C"
00734     "208D882308510C75E729D08C779CA99D5A685E78D5628094AD137BAA635B7FC0F492C48A9C"
00735     "DBE63209C8231455012EB3E830B5B2A79ACD8FEA8016243EBC85BF5D6F46A48FE013D2B3B7"
00736     "89BC5F743200BCDE03995BB2B6A640CFB099788E380B4E01D75409A8D8B3887DF2B1CD3496"
00737     "0091653EEA6C52EDD745B9363BFFF666891D9C8BF511C3C07D38F49DA2892DCCEC81E1722F"
00738     "6EACB3214E3335C93E6141AB94E5EC31BABF8108F6BEBC3E60B1BFE37579B4D5DC8B77A347"
```

```
00739     "940CC1F6BFB5B46097B1EEEC4C354159BB3475E05FAB6BDE5672014D9489CB70DDF537F720"
00740     "9BB9EBF1FC6B8B94564AAAD5ADDD83CE6E51EFCF73DC6080D738C4FF1CBC87ED420A0B92FA"
00741     "459AD7BE58789F0A191D149F88173184A22874DF6D39DC1BCD4413648B178ECB03F8358547"
00742     "A68DE7B672BE9BA1FFC8BA392F8A58ED2806155C00F86B7669BEE4220D420",  // PRN 49
00743     "97051FC67ACA30E8AEE73D3A8CF38BB13524D4E0EBD9BE68398C7C16227CABB1D0B0A0ABE7"
00744     "B6384ABA02905BA0C3C7363599D059C7B4C99DB165CD14FA12FA7912449CA7DD5E346D8010"
00745     "C85A757382270DAD15BA3CE36A76EF55F81A1E80BF366B37FE3A88EC722028C25E234E6240"
00746     "40450A99CD808F942568AA7133981D72E7F2928894670AD5399482DF1B90E7E64062F830B7"
00747     "36C79C30F36281495C76699CD48404673FA334F042F9E0E67DD7F3853BF71ABEAF6A9A5546"
00748     "855E840CE42B224D8F6490C6CE5FC02EBAF4FFC390107058F54CD635D4A7F2878099C1EF49"
00749     "5750E6921BE2F39AD808C4210F287319F811A254CEF8CF153FC50AB2F3D694A530949E5F57"
00750     "8D075DB96DDCF2BB90ED3DE09D9CA8E08662FD8982741DE1CE0A6B64C3D3D5004B5C04B2B0"
00751     "DFD976A20FACC94D1762D41EE03B40D2CF367612812EF4CC41D1BFE9CEB51AE3A22AF1BE7B"
00752     "85A057D3048D0E73FA0FDAF1119EFD76F0A41BE63128B22D64A5553E9549D411483BBCA148"
00753     "3EF30CF6A6D317AD2C7973EFA6D4C1121F703D2F48FCDA3177AD450D75D2A28D2C244AEA13"
00754     "F0E60AEED8ACBAB444D400DF5E280DB799B2D9A984DF1E2567D39D1DE58EF78CA6B4D8BC17"
00755     "2B07DCB02D156CA96EEFAC69E556CFCE0AAB617C7FBB8C34871C1D35E74B7BD307D3F2E424"
00756     "C7A9AD676A1A69E0FE735EA50887A1DFAE6CA2FE4460FC7EF323ADE493020"  // PRN 50
00757 };
00758
00759 constexpr size_t GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH = 1023;
00760 constexpr char GALILEO_E1_C_PRIMARY_CODE[GALILEO_E1_NUMBER_OF_CODES][1024] = {
00761     "B39340CA1C817D81EF4FAE4E95BF3504A7709089FB48560E9E3EF802180E85EB2194E05902"
00762     "C6C4C52021FEB7EC64FD416BCEBC8E39D64A4B5EE345291911AB8204A888C25B1CD3D9342A"
00763     "56C538636D3EAB957037D09E879AE5F3A39834FBB84A3D8D5090D7814246B62E9CA68533D2"
00764     "EC403B4FB9488467FF9758B0D15A8CEF89187A1D5897880040B6C3C5244E85A2AD14BCF2F5"
00765     "ABC44A7B1D4A87E8BDA05766218773ED4F70F8D1D07CBB1E8CA6298E64EE6DC5886D37495B"
00766     "A2EDB3E0B0B68AD9F300310B88898DDEEFD484538C31A9BCAA76ECAD0C16607D32189058B0"
00767     "862EE9D70CEA9D304755CE8037BA4C46C2573181748A212E4B2BDD04F9BC240518273DC17C"
00768     "BAFF21A03E9120FA7DCA18D56DD1D9A7E510C90CF219104385F531F2EFAFD185ECB6B911F9"
00769     "B7809D98D86F15516FFDDBE9BD1CF8662EB777C3F94EA3F962D7B79449FAAD39935429E92C"
00770     "AE5637E9BCF4E94D413D27934952409AB536BE4055AFBC4330CD1E4B5509EFE5F8EFC9ECBE"
00771     "9EF377DE7E37C479BB9D3EE7745E4609B0A6D2C5D92EB3C9E2278C1F2221FF907596AA5E09"
00772     "6ACF8990EBA907E43AD320F8019CB6355A2BA8670EE5A4F463E8E56F8F1D3E7F4922510FB6"
00773     "68E32C4CF23AD8496399638B095B47833E0CBB34977EB3E4242EAF870D86660D6A73F83E45"
00774     "D6E8A41EDCA3815079649544597C5C43B6C93FEBAD5700D22EDAF431FD340",  // PRN 01
00775     "A64F94BB47BD4033C76D4924305907EC1F618B43C7535F3CFC093E5AF5DDD5C4339F3BB6D8"
00776     "35B5C2C2053CD3D5693368D4E1A7CAC59425D1FD96809C67285CFD3FC05B01053CB0773221"
00777     "D7205778022F487BF99D1650566BE287FD7AE882AA8E8F52E5D4E3C0C2F971C9FF70AA3786"
00778     "91EBD8ADE45CF213822D09FD05243F9726F6C69893845E57C37A7643E16B770E26F431FF69"
00779     "D437271905D270EB85D8D229D7D87662121F0BEEB1E895ED9589A9CF5833408A04197AC902"
00780     "5D8570AD9B75DB7E192EA0A089504996E9DC652975D83633619CFF80667D8B519536B34752"
00781     "48BA8213C8A4C66DE69B4B3774BF9142425C57F34A27B1E288119E3FFCC6AF6A21087F9394"
00782     "F09DDFBD42F32D059B8CD4104A519BA640765D5CDE490E62F10E695FBFD33CBC9D2208A532"
00783     "C8EC25DA28B8CC1B6850AB43D9B5C00B6E74B7A148791AB07B328D347058C7E6233E18C5ED"
00784     "172C9F9E9ACF29D913E2A1614BFC0893D4967ED033B2B9AE6B51F908F1CED57C14FEEA85CD"
00785     "4D9711216BE7F79FA6721B7DCCA033C80127AC6E5FCF58EB4005EC24CB4886D787355362D5"
00786     "E7031B9B2AC2A86D730AD734181E723A811FF510A4DF868001973FE83288D78E6F9B9441DA"
00787     "F5BE2974A2848FD917C3BCD346A431922246EC852E4AAD467E60C15D61DD3BF4A207BB57DB"
00788     "45DCADEFE3210BE74B9DACC918A394469F2E2C95AD1E211947948FE24F5E4",  // PRN 02
00789     "FD1F6976002C39C87187C44E3D224ED4DF0B67750105944C651A5E57798F168A136AC0FB59"
00790     "79C4E847A82B20A2E6C45DB42EF2B930A80D3257BCCC53EDA966F5DCD9AD47CFB226EED9B6"
00791     "2A874E9F6404D4087798A1005F4131171D3A47907A3CD602B83DABE094D2CB031867DF4595"
00792     "F3ED59FD8C4D76EDEEE59E422CE5C7D0A5F720BE94FA24DF05F758348EADD5EFE9197C6BB2"
00793     "292E2B14DB8C6DB24AA94C5FF0F5106D2B566058D32C58B63A150784F7B02478D9973DD4CF"
00794     "D2E84059AE0F4F1320754B7EE83F04A51C67EFFC2EB1C301C0C58DBAEBE95474E3484A7650"
00795     "0103C14C40BB0B7D3A04D8BDABB605C1EF9FD4A6564934DEC50BD5878243AEE80F9796EED7"
00796     "0CE1B1E8B55725DF76472D12D4A7A487989F42E6705818B1F7E149E97153A7B05A82FA3FBE"
00797     "51763E61171A4E12931472E94CCBA74CC09483DF93623FC60945070FDDF3A00B561650427E"
00798     "4BD64D675B1EB398B35EF057A66FD0B48EDBABBDCD57C32ABAE46F5CDD0CB1FCF177652582"
00799     "36F3DE40BD5D0A3C5C978D81DEB07367AB20B2CAA9834B9576161C4F20FB9C184A01DC9021"
00800     "A4E92B71333354E05BBEA9015E5AC4C66312E8B79F0B92279AC7EF1936BCC30802B83DB3D1"
00801     "13BEF64452CAD7ACF6674FDA44023A661019841A101BE80FDA4E3210AE774E433A9ABD97F2"
00802     "755259AECE21F7A8C3B1A3D471F874D2EEC85B9B21BC0C2E2EC9016F847C6",  // PRN 03
00803     "EE38BAF6F61704B01509B5210A0534E4702F93190C392E749869B5572BB7AC4D7120E2BECD"
00804     "6618CD376C4C1B4965F7D9D73400824E88A5C7B5B66BA88C3E0065F9628A9AC6B91A188219"
00805     "2FC553E3140349934D20698C9F291B5370948AF6CC90C837B9F3607F13CAFD492CEF172337"
00806     "6E6A5B813A56301B88A8799519CB7646F33F91C44CDBE7F768D7DD9B323A5002D2F784C410"
00807     "1AF90D6E4C5ADE7D085C79E827D43E10DF63AC70BCDF13DCE0471B487C5ECB752B9C3E20F7"
00808     "5DBD243790C91355ADFD7199081BFEA03D80E82445EC2831FB5014B85EFC2A52748A8ABFAC"
00809     "1BA3904E178DFBAB26C1750228C9A031104F58BB3B91905EDB9EADF7B0F6DF22ACEB0DE944"
00810     "E277809D77507D18EAEDAA1767697398421115D04AB2EBFC466E99F0AA540482A49C6AC8FF"
00811     "95E3F962734B03EF39873A93B70470B46FFFDFDC15C89F8FE2F4637B59F9BF9C5752D9F8AE"
00812     "7EA75D1EAF1C22CA27E5D5C9499624105D61BE2A691F9194D277414532A5E6C63875F7F20D"
00813     "D13C6EE73B0C3568392B14A5042843926472ABA343D2C427792199B543BE1D43A178FAA7EC"
00814     "F53B98AB7528D8E1B8B82C52D973CA0427636505837F94284E8D6B4F496FC5A48B7958D468"
00815     "1DA00651B8A7BC56EC859C071E4396A05F33588B8087EFE9635E565E6B5A8A70DA70F50ECA"
00816     "D1A85E6E36FF07B4FB3B9119EDE0B611CFA91D9D4C58C1F4815B07B9EB1DE",  // PRN 04
00817     "CD37D0FB0043D03444A939E93676B9DAF5F2D19A2615E3D97D624E62ACAC8098099FDB9A5A"
00818     "2F4B3ACF20F75B6807A5A3F157C2C0F479158F4A10FB4972855F3AE2FDCBDEEC00A4D470AA"
00819     "DF5F5E571818AD6E872D897E2DDC402006965ADF16582B1E06B1861BF7D0C7E7BA491C79E8"
00820     "6224AF6B246317F725FA74DD8376D63D7993FE2F2BBBB2F1DA9238C6F3FFCAEC50FF61E645"
00821     "FADEB6E03F883892C42CCCF904708B123C9271A670D4DCFCD602951D12F5213937CA2C05AD"
00822     "DE9EE3908E99AAE8DA31951C36D36D671CD7BF15DF60B707F00BF6EBBE5476926D015628A8"
00823     "5758BFF35C4AC540F39E761B2ED3CA9116E8680E28BC3870558E0F69345CC6AB3AD160E9F2B"
00824     "C4D6047A1934E15D3D7A242A296333C09296981BBF3B8577E4B8ED2A3624866111F6638F89"
00825     "55431195B60C5C089F9897DDF0D34A3DC627CE337AC8128C28B63A394908E4C083BCC4522D"
```

```
00826     "B8CE5720C45EF76B2716225E53405FCAAAA72AC198226575D5225195F106C1249E4B87AC05"
00827     "287A3ABE6C51A2A41E07F56ECDC46E989A8568D35669B525A6FFCA90DC91D3013967F6A5F4"
00828     "C022FFCC17751B68FB0D8F16FC9229851DFDCC060838F923BD44C1AD70A993E8EBAC1667DA"
00829     "80F91B66F8F5B375D35275188E3C7702C2312CEAC5B20D67BB34400401BDF1DBFE79DFA0EB"
00830     "73F173A04807215DA5CE8E1D28F2126424C3DB44ADCD7A961260FDBCAB31E",  // PRN 05
00831     "CAA02DD19DB9C721EB35AB7D64B8A387796427242698A47D832C3F1AD4DDA0B5926FFCE931"
00832     "9EEEDA1565ECB0FA1EEDB424414120AAE8CFD0BE88D4D248899A0BCE31F9BEE7A4DC4DB3C3"
00833     "B10444FAD6ADCCE28F0EDF7B808536ACF5EB05AADAE92693EE02C9512B3EEF000844BA35E2"
00834     "4620A2E8935354B8432C07C8FD615534BCFD0D8E3B572BF2CF06AD343997590FE8B244A32B"
00835     "BE69125B5D7C5E513A493724EEA8DA6CB0FFF3ACF1C5085A8120694CBC40FAE1A6326FD714"
00836     "87CC3BE7C10A34315CDFFA8C618B68EA93D330945586B080381F0076351B888087F56B969E"
00837     "6D6A311AE03CC79FF6861E715C9DA9AEE751F1220661581C75DCEC0515A1C9259B9CF8E944"
00838     "CEC4B1754E5809E985D6F43FE45710893242ADE0D3B84F1E1942B7A95648611595FED13F54"
00839     "6CA11DB8E5A55A3C3C78C3793C6689E1B3AFB5F67526A480DF923A586A779F94A09CF96359"
00840     "4FF4B0A387876EBB3E8FAB888C97F6773E7F0317B038E47DD7D109545BB07263B1AA84284B"
00841     "86E47FFB9784A171D101E7B0A6D38BCAE7E63D827C999BF551728FFC642EE690B01D486CB6"
00842     "EBEEB9D5C888112589EA5CBC9BDF49E675965223416D6DA02D2333BFD4614706BF13373973"
00843     "207C849A0DE41EBA137FDF79A1EB25D74E30CF60B577C2787DF04740BA8CADE3F9DA55D3F0"
00844     "084F02809E37543239E0A71E99751EEB21CB3B414882441934A4868CBA9276",  // PRN 06
00845     "FB227530F82BD527E648619E532D7646A5ABBD15DB91A6E7033DFECCC65D095A3D83AB77ED"
00846     "D2F3FEC52659CB3AD1BEB009D7A1C9BFB544291EC1C67B75DD6DAB06E70C32C714983139DE"
00847     "4A41EE07B4F3C03BF566558484F19A3BB674B6795F0D8537BC31BC8D7A38B2FF1B2EC8B785"
00848     "39B2251D0E385DE484B05A411477681A3AE7527AC98BC2943AF1CF7F09ACF2DDE4530AE896"
00849     "BDE1266FE916E833A1C0CAA2B2D2F5985AD47B2D0D1D3AFB6E50D4B3DA7DEEC4385E6CA8FE"
00850     "22760F92807AC55556AAF7973E8016ADFD43A3919088B768351B1057498D2D668D7C1E8C63"
00851     "438055FDF7D36C5E7DF02FCAFCBD9291A2149E7B429B3202D329E47CED51EA5771772E308C"
00852     "5BEBA7B934597540D83DBEC6C3BC61A96EA4CB2D7530D9D760AA9403338CD95B829F17547C"
00853     "5A90D161F7B8CE0037EBF403C91C0D0C70C589BA87CAE8DF26CF14281E235A686CCD10E2D5"
00854     "20A76265C4C2780EDFD0705E89EFE3C953FE760DE45A8CF1F2D3F36DE3164D5BC2CF322042"
00855     "28ADD7C182EC55F1158AFA9358BE179C722ADAF1D0BF1306A0B56218857FC5C21001499F61"
00856     "E273442281E585B3E6DCE148AA97B6622B23BDAECF983BF186F1B34962764758AC3C20C840"
00857     "36061D49CA33B3C3FCDF03F47F7E53B940DBB6E1E4A26702A118E525A9A0EC229085C925D1"
00858     "33750ED0B200CB28A113289DE143D1D5839D2AF8B0525E0027F34FF32106A",  // PRN 07
00859     "9E5DA18A19514CCC849E9697AE4BD1B317BB34927D0461A96A7AF4A5D6C13107FFB9DE38C5"
00860     "E8CB7C5682827F57D94ED2E77D36F9F1CB05E4C2C62B1DE254C7B1CB236FC4ED70BF8DD1F4"
00861     "3AC773C16A37392B895F8B157578C477C85E53FA7CA58BE70D9187AF5F7A18D5A1E5642335"
00862     "E46C2F8F4691AEEE6A9692E21B9668E2C083D9F45C2DB3E991588BA87A0A23808732EE39E8"
00863     "B3C876BE79227C782F07EE3FB3086AF913D71D71910A0F56D62B5DE5E224F7856A42A4A1B2"
00864     "AFE380827BE86E381FCE486FD08A91B22BD91D09615F417E178C5593E41B0917E075133960"
00865     "AD28B4DD4096D1E84BEF1363098DDE92C29CD508C40BA7E785F46C1E0DC72E729D394911DA"
00866     "919EA6F94D14567FFADC61CEB8DCA2821B1CF048477E2433E9DC718DE618EDEEF302CDCB5D"
00867     "E472656D6687DC41EA34C2BB4DF1CA08DCB933BE3EF4B419158BA0B68AE82A64ADD5855921"
00868     "4FD88A4CB34D99F646310697DA982C2FD4EE069DC1CB102125C34A89AB20F17B6EF648A834"
00869     "627320410FF6881C7919AE4E71CBAE5F8200E523934D84BFA897C44B89B9BC6BC0129F7F97"
00870     "EE0EC049BA1AFD67D00CD624A75FF5A30514399BE4801CED057B498B9DBBF0EB9944295D5B"
00871     "6AE968C4B8BBD2B9A9E17A3039C5FA35A0D30AA54CA426C58353943DDDD3FD185895C0DAEE"
00872     "950455FC131F520B46AE118C7406D0A72BE6127C5307730AD441B6FC3D1E0",  // PRN 08
00873     "8589F8396F5B1C54CAF2B17D4C152CEF347E66EC7903C878F2823D4ADB9E7CCFAFEBB926B7"
00874     "EEB4AE1BECA339A027CE8EF997957532FA871F356E0326ECE0BCE3399F81179BF78C5C7D13"
00875     "5018ABC340C0BE58D3063DD7CDA4C1918A0187BACF830C8B6900D43B62E04DF6E831CFEFA1"
00876     "3BDB5E873A527F24327C95DB4BBDB65C81A20F959F828F5DAE4DC13E5CAC7417EE089401FB"
00877     "497ABE10144E28EA383E61D4A9B63B618AA7CEA4588B2911EC581F506062B05E7BEF723A5A"
00878     "465C9FBE70E313753BDE3102845A79A206BF7D996F49A21752D534B73EE83B48C1A225F85F"
00879     "5103DDB9B6B8380F61AAF26E5CA643EB62EAF58AFEE0D3494E4F7A4F642A3454F4F56A406A"
00880     "264148FF5DAC9DF5F151C12E89ED9D4FDCC04EC5F0022DF8CBAF3CBC67CED2853FB4F8C589"
00881     "4C96CD00550950E7EA2A26C80A72DF533270A0E23EDBAA4D0BE935D62CC885E1CCE653D66C"
00882     "51E49C43952042E1B2D043BDA1CFFC1E98A3F806EB587A4EC9AE299BD838C68B9BBF7C420C"
00883     "12B23AA2793FA0248C932A91BCDD641DCB38F0B2D7187D8986928DF4602B381BA13B263291"
00884     "134628FC91C8EDE92594B39650B877D9A91DAAA05295457DFB2C5D8207BBCDFE16AC5B9360"
00885     "0E33BC970B38E18808B1A732889320352B524B109560136E605D32784CA01F8B11D077C81E"
00886     "AD6B7A5741C82D76CEEF764FD07E361D531B75106AF1572AD1375B2BBAB68",  // PRN 09
00887     "A3E17A4CAD2ABE76E32D18501899F8D60D293BB1AC3ADB64F81148AF56741790F87F8B7A2D"
00888     "9A6E7645EA50B75514C394508884CBF9E320B24D41D8246EB3C163B9101240776C312DB63C"
00889     "33889E3C12184358504 71C454486DF7FF4D2DC0AAA14980F394CC8EB7B828A60C53A2FEC33"
00890     "15BEAEB30045B3E65006C6EBB23B47A8A069EAD45E32E771B9C467B4359EBB681AB48C891A"
00891     "BB796544169178203BCC4BC6B4A278DCEFACE5E9385C059346A23DCCA001FC9E47CFEED4BC"
00892     "BDD947B12A3F7E5FF8B9372D9497EE1A508D8BD3392BF3CFAD58F0191B18F6A300FF9CB8D9"
00893     "14FDF37B48BF24C2C5CA76ABDFCCF833D51D48FC90E06E7B972944BCBAD169232A8429B610"
00894     "0BA562F7F3C55A625A1870A7C7D7BC9BD4C4783278CD95D07F89E8010E78876547F9AEC443"
00895     "22B0029A922B2922634ECCF2BBB47BF87909C494049550F1E6D03BB5354DEA7E777F499D2D"
00896     "6239BFA5C1CFA536F8CB16F4DB9EAD96F83A4AD34AE2C6893ECD6994C89E7F4FE426D95A18"
00897     "F93B88CB357996B8E5A34C43533EDB1F28A8162FCBEF03704FCCCD80C32874F345D34E81EE"
00898     "813DF5CC9B9C299362F8443AABE91BD0EAB9746E431804B6129FD32916303A570323FA121F"
00899     "7AEB2829F2A50A82CACCF6D273FFBD7AC6FFC5807771D216F50742F7091946F91460115989"
00900     "C87E8BBBC8402B4C8B95C102CAB53843D581FA9F16C0ECCE8944E5FC4BF4C",  // PRN 10
00901     "9D7B1CF0029261D65AE1F021DAFA81CF1673C9E0B47FF2C37D1B1AF46E7A91BC5E529C8F93"
00902     "EE3BC74E92B2743AAB1EDE16A6523B5B8A591C617C1FD0150E63F3B7EF0494162437B0FD55"
00903     "5A83A3BDB519B3BB209EF7924D6BCDE5992BA6248690442E72CD5EB64B4C3D3F7DA339108A"
00904     "18B61AD88ABE87BB7C85A3A352D7B882FD683B2637A17A2D9CB0B7F41456DCFA66D62913F1"
00905     "45600BAAEEE7EFA5071C3C9E6FDD0A6779A737071FA6965978CBC89776386B108DD7216FCE"
00906     "962FA87A26B29FE0E732309C0124B0C1E99E5642E5EAE670005B078C097D16C58B8923633C"
00907     "18FDB0E8FF8C4610B789387ACB5A2DD0B6AE7E0DF43A6A9E8C3B89C7E5D628D59759C58D07"
00908     "E0687812AEDAEEDBC63B4FEE8524D10E4B467696957E6791C1E94B13CADCD0ED60752C2DB1"
00909     "B65E035EA72F89FC679138D3609FD2A30E4DD1A946418253C67AA69B07EBB95D4973F562CE"
00910     "3773430007A6DB77271D5F2B342CC5E76E115178F9C7B1600554F5C794961BAE81A5E9B621"
00911     "BA17851008BED9B556E461A553FE9BE00A40891750E4EA4B475216283B530CB8D479DC70B0"
00912     "26E07889229F6017552AB9E01EDE6703FD1E2D59AF0B71E0F1DC9A42ACC5823324BEFC52CA"
```

```
00913     "0DCD25FE8B10C999152AA3676A30602D3506F78751477033DB7AB1A2EDC21A6FE51273B6B2"
00914     "890088703CEFE74F9EA89881896E5BE124B1FC9430B92F0C0568F5A068A80",  // PRN 11
00915     "F23088E3EAA0A6BA04D0633AAFE85203E8B1829223FA6B730F6DEE6799B521F2E8323B8793"
00916     "D0F7F2BB9305B3EF4F5B4F1CB822836E4D92C8E4928A851BCE688329DECA6F7285DCC85195"
00917     "E5BDA3B503B8AEE6F1CD7FBB158444E7DE8BF6A9A3CDA311787755A827BCAD3DA5621908EA"
00918     "913C0316B9B52BFB07ADADEFF17D3766BB450DD71328A0353B09DC24DE93CF83A2E5F98BA9"
00919     "D612187B601157D6B140E675228B58C9398618C3BF0D11A226E489366102B9C35A916653F0"
00920     "DB36711ACBA5F32B327F5789F3EF48A338E4676F4BC2C6A1308597171903D2AA299CE7E523"
00921     "C2ABE4B15AA4FC48954187E0097583EB099419047244B4931326E5923B6313DE08423DB008"
00922     "66374ABBF5C31A00542CB97CDFB8F71046AA2A6DBFD7E1A71C068ED70E8D7C3268EA3E0EEF"
00923     "2262BD7991B6C59FF471F73A4E85F4FA015E164F9C15FE0AA5F4772BF2D62B26D3EAA25CE8"
00924     "3EAEC5EB3577CA83A68168FB64C40A7A155905CBA6E64159E55EBC928D125E55165C639F54"
00925     "5B0071EE3CF1A3F58B4994BB4BF50C2B24F2E06E4ADC90BC1C0954A257D88444347AAECF13"
00926     "6C15242633463DCF984BB6736666E38F1A45150B1B7D1C31DE06EB9C2F4097E9D9B4D21EBC"
00927     "9F3A918000DE2449DCB3F5FDDC3C773A645DF560F7E013E847E2356D33EFF1E215782638F5"
00928     "8034B09F4739F98915BFB0B1DC124681492F58021670D03CBF5E8F962351E",  // PRN 12
00929     "EB07F9EDF03596ADC2A3B7EB6DB1CFC911E9A4C42336A57309F7B6C3389282E557D94BCC71"
00930     "827D7C5737B1C530D2A087E3F507242F3DA5BD1BBCA4DF8B78BEEC1DBF7EBB2EA1CF1DFA79"
00931     "E60785BAFA23658490C9A64AC61C45779DFAFC6C55CB5C9FE457BF47E45A3FEF092E178ED4"
00932     "495C0357B459E95AAC82132FF1C8044F4EC84EB882DC195D9CE996B1CCF523098E9E1A57C3"
00933     "7C2E2D0ACB0EAA34B0B56FE5A0747130B1E75AA923F6F94C0D024A7FCD22E7A4ED8B201966"
00934     "C417AE864420767AB3223BFF56C64D4F8F557DD950F7C50D9A39AB2C742CE686C8F92B3571"
00935     "1904C600A9D4D3DD83F3DF1ED7DB8042C76B0B7D5D9BCD6E0B5524184BF99D8D0B4F14967F"
00936     "A48A93A2F44E2275ED7E59F3991EFB0CBF2E26AC1F8D9A41AAE4563179254BA37028867E68"
00937     "C8179454B8B71FAB49DBD1F889104CFB64C8121151364BDB64BAF854B0DA22B8620BD7EE3D"
00938     "4302A88A115F8BFBA649CAA9EE7EF5BC95CFAB26503A9D26033370A4EF3CB8A5D094C63305"
00939     "A833387B4F8371C6FE1987514BB458C571E6CB5DF5FC900631652D3FA4444F8F1F03122043"
00940     "40FDB2092F709FDC51D2680753131ABC33712B4F1067EA1CC87C40B281E69209EDEC42C22A"
00941     "88950E9C1CE8130DA9291897BF2D8D1D106911743E7A9DA36220FA90A02A34EB0B28543217"
00942     "839374EBE79F40B3B612236C902E4CD05CE2E1C07F3DA10E2AEE8E387494E",  // PRN 13
00943     "E9D537A821DEDE526B441BA4252785779B54DE76F82747F8607B8952DF990F268C039CC792"
00944     "883B1C76C297D81C6C0CF17DA8BA2C71110B16741728725839D33B5942BC0A5614A3650675"
00945     "FDA5D70F29154A429A42819D6EDE324C64596F93E84CC9B2C9DA3717AA6DFFCD03B75AC965"
00946     "43020A9F2024620353E1364E4320FD44933799FFF083E73F5D20B83BF77EC2247964ECE442"
00947     "C3213DE99026F8FAF0E96302EC60067EA38C5CA0CD989475205FA38869E349FC7F79EB81F8"
00948     "457CA3D1A875A8D166C96EBAF1F39C88815E2258EA1A14943298DA39EB9B738AAA4E0035F9"
00949     "567A0A9D572785594496316D56EB3D39E1F3F243D4F16111E194FC537A635FAEB2FB4401CA"
00950     "A9EE0091CF3CB28B366CB5446A6D3B10AB86B4B1A0714D107FCCBBD50EAE520D56A1161E03"
00951     "849192F5096346FBE5150B6D04025A564A43A3D22BD4B7E10DD4061CE20FA2ECDD36F66BAA"
00952     "D7EA96CDBAA0F063B814707718F47278F8570F77F3B15799D0E354CCA50DAA38C31C746B17"
00953     "482297D9C089FF379454FCCCB8730D89B1462AD95426370AC37DE50B775B952663B97AFBC4"
00954     "03F6F729BB9CC1D21DD89EE78AF09DF8558F7E68B3711A7D9075DD4754174802F52CB9683F"
00955     "FE746471C7E543FF388D024327D1866CC5CA6775C58A14D70A3ECCD3EFAB52F9AE6CCE1467"
00956     "66A8419FB546E39EB604F43B15AB88C72741F8C7D0A7FE2F462D360676D6E",  // PRN 14
00957     "D79D916241BBE52B61BE8210A02543F75A47032E9C0CC128524A675E94D8F79A69B6842B0C"
00958     "5CFF5C1AC98D2085299BDBAEA67A41C724CA36B6275A80D377DC3A6EB4C8D0B6B88241334A"
00959     "95300B53FFB546163D2889D7C85F1D1397924F126DA76085BEF131A65C7DDF60DDF4086BD3"
00960     "3B44D25025D689FF41E0C256EA12F4353D9E722EE37907AA8BED0A5A606333A031AC6B9A16"
00961     "614250916759B72FE6C1828BC6C1966C9EBCD51413A77F41F808BCA2534AC49DB1D32D3787"
00962     "8DF5CC0BEFCC099C56CAF50D8B92E7CE616AA026EA1D81DC7ABC17C4705F9B57A0F99FA749"
00963     "F30F93DFA982A083EAE6582C8461A11ABA74B11663ED7D66EB4F8DE14F090EB1CA6D8D81CB"
00964     "6B063A391FD354DCEAF7DB71C277D0E92B4B463873DCBEFFB698BDCA17F80845EFD5F0FF15"
00965     "0ADDC9D7797E21E4279B54BDD4B7C9D403D9FA6101604B79AC377780A546149971408294 23"
00966     "13CF74AD1147CD10571A31D82871B6B3A055D50C6CDA4BDDF3871F41EFDAEBE8ABB995344D"
00967     "B6366E35C6E506907AD7FC76632F99124A58A32C86360FD6DDBF50324D86694518AC44F1FA"
00968     "19662C0EF0C0860811B5B976A96EC2A1449E53A7E4A07923E9F85794F228E441D92903922E"
00969     "5783F2FA21C677251B6B8DB02AC2E242C0C8652E0C17C9E3858E52DE78DC712B2DD5D2AF9A"
00970     "42DB2E2BEB3FB6E0FFF13DB9A1E02C8F84FCEF3F7C4D2DDC09F2A2813E8C2",  // PRN 15
00971     "F8E2DACDD88277D482951555C657B3E3C5DB79E5A43500F7A2C8B30C854DBE611FAC1087FA"
00972     "03D439AC4635D39211E234B82A91248DEE5D4FE67A02D5AE25C676E64C4843E419EBB3C4D8"
00973     "1FB606B9CA0836F8207CD19D106C0E287EFD8F8DB5C1A3A22886C2765FED26B5189153657B"
00974     "7C47D5590F11C6340067B800669B05A0849BCD2005DFEE6DF95833C9E94328D72F931D69CF"
00975     "BB2BDA81AC83DD660B3B17D2BA4023491DED324FC4F22510ECA4A5194B1245F4F3FE334DA9"
00976     "C1E6BF83A3FB30897BE54C688D2A7C5845F425866F25DD0A9852BA6DAAF8437DD80BCC72B3"
00977     "E258A906DE079A2D33EC5C5F6927503BA13158305DFFD3F86345524394151AA557D6242060"
00978     "F276BB6BB25586F632942ACF5E0883CD3F8393688F360323A000B82BD89414E9C807994B02"
00979     "34D730BC6D7CD0A2BF75D9F510786E83EE98D4CACF20EFF86EE9C38B8D52455D8A694B689F"
00980     "0D9A632E7A6AC6675E190A12ADD716D2C6322657B878FA97267C1BA4631584356768EBBD1F"
00981     "13FD2F37EBDCD1DF96FB943942E8A5188666235B455BE2F770C9759A8F070971CBA4978974"
00982     "4FD2F64DC4DC6E003B3F9BEC7617C7EEDF6BACA94D374400499CA6813C90A03DFE2C537261"
00983     "DA93A1C0F6D8BA93D1EB5FB17255DF28B78737582FD675D056A4C474A71CA8EF0D77BAEBE5"
00984     "637711AEA3FF2B014700448C3D74E3DF264D773360F45CCC3342987169C9A",  // PRN 16
00985     "94741D7F05B0CA50908E6BC14801A28E353551F01769451B1482FAD0043D5C72331246D9AC"
00986     "3344F0FA2E28FD00E86B38F5E0452F46CA111E92D01B37E966455DF1374883DB8B055C4DF2"
00987     "5B42182280F86D0D825C096018D2949B4BFCEB7BB2C8A5BFA2C79E27F11A7F9B43A50AF928"
00988     "D81FA95CEC86A114222B99786072311025672AB04B2593C5AF50100B71D052AE268FBA992B"
00989     "F7868E58EFCD07A24D2111774A36115C1C527B5192EA955722EAE849EF83817FE8595C96EA"
00990     "2D76FECF6476D89F65A262D94B3F5E89A5DE8B1A7333EFCDFDED17FE1CCADEBA0D1E7B73E6"
00991     "7491B413A862E34A308D5C211787E6ED8683C6E1DDEB8EE2D281166C03E7A72D7D7BD8B878"
00992     "D07D2216C21B855CCDA76B7B75DD1B2CB876E59F91F040D42B97050043499DCFFC65AF803E"
00993     "2F7455C9669DD9896FE1F62227936DF905835A644D31130A39479DE75B4DC4361E41202D51"
00994     "D50E0E4B4B218AF7F5CAF264DCD060C296E777DF1EED6AE8147E9B6CA73184C345FBDD89DE"
00995     "4A999C42AB4681D9EA3B86DD75031A33DCDC807F8FB14EE0CE61B16068AF01CCE7378C9D96"
00996     "5943476AD21A469D8B0CAE15BA8FE04971FE1EC61D3AAD3386DF71B33FD0B4F324F3DA518F"
00997     "0CC0353182B3D76CF4EF5AB150FB9E74C28234CB3D907AC81CB6D3B99D510B481E1F0423D6"
00998     "F4987F5517ABBBEEC07F46AECEBA5F15D91AEB0FE91490E91F739D465225C",  // PRN 17
00999     "839A01464B473A64A3D1EA24EB363EAAA590F4BD0E4492FEC4E3D4DB5883E4873BBA17595F"
```

```
01000     "F48134893F16F5C4A43659C46484A268C3303B2DC345E8C98FBBA6D06946F997074AE15680"
01001     "EC9423D6464585D98804B3541662E183F6540503BEC204749D58E3DB9ECF11C80CD3A38F8D"
01002     "66FFE6CC8A003BDD35F547E5039DE9A21F70A8A07B2DD89B68E43B42C2E021A11909817C54"
01003     "3F839E6862268E38DCE712B4D49C39A5035F3D6BA19AE028AE70CCF557720794FEF6442999"
01004     "E740CD6AFE6235F165515FDC24AB6F578DB2549C8065E008577FCF8B8DD8A3BA679BABBC9A"
01005     "747A4E2DABD91501424E4191097E689A741EB6644A771CABDBFE6B74ED3ED171DF8DE641C1"
01006     "D42213B9D0F8CAD1E11FF63670F5587F1FB7FF92276AB48F31751E7A591AF4F0966F390988"
01007     "3EE6015639671BDC3D1378750F66F5DD165912CFF1A54ED463905404EB7D3412EE2B0F0D9E"
01008     "6B99EC81678ABCD1789BD8F1D72D3DF8754A16DC2106B83B325807E27BCBD22A25DAC32F27"
01009     "EACAB6A4CB6CBA4CC90D5302BE5E9827B7AB48BB696B2902975C48B3A4BA4630B14E0FD8A0"
01010     "50B0718C2829371BEC597387172B0B3192EF958BD1F7977EF9A3A6C80D53BC961315F97B71"
01011     "4253B9731A017BE2CA1D43024F75E26BBE989C4D514D01538956FE4B90BE17B3407B55BD08"
01012     "BA50FA807D0E448B7CAC65EB3FF856772A933F0C5F3E6F41E051015C6F9B8",  // PRN 18
01013     "BDA2B72F0BB0265269F198207FB061DA29DE43E30847E7C062A581A7EB53491EA51B51EDD3"
01014     "6F991D15AF89AB53198537988350FD5FDF8E003019BE115840B9BA55C238C3CBC72C0E24E2"
01015     "5090A3D6A59BEA9FED0FAC9EAD40451A95649638FE0BB0F8FFE61AF5B9A8AB84BE84C65EA1"
01016     "E12E9F6650ADB59A824E608E80D1FC3AC19F418169B3879CC946165511D5AA280AE644AF36"
01017     "0C42F7A3EEDF27E368E46480E3353E67F536E02B33505341BAF3941069567B723D7C125C8F"
01018     "066F9A6255436AAFDCAA8C554FDAFB0A9AAD91F1263DC62EF91A748FFB29F57E325D65A38E"
01019     "CB4F2851923DC6E9B7296064148A9BA2D938116266C597D9E1F11A46BE0EF526225BE750F0"
01020     "F3E5B0AEB7DC2140FA3A48B7238D0F5A872000782CB6F7751443EC6A1B7FA1ED02B9ABCD1C"
01021     "1DE4FC85E9B405C7851913C60F85582B1529276AD475AE52BD8115B6E73A53506E7A0244E1"
01022     "C29BCEF4CF20CFDF883392BB3990BE2A11B3213B68EC4A166C77D724CFAEBDC34C45ED0984"
01023     "8A994BCE1FF6A9BB80C7F5CA8FD44D3FDF8DEC8BA6552C234EF8DC52382D52D2B01BB23404"
01024     "FC453725C7C9269A785FE09C712D4ADE7072B66295CA0C6405D9859E134FBBD3737F2956DD"
01025     "1D718A9F8242CE95BDB1E49F265EBF19976BC46E29F7DE0EE5C89A43AF2E107588A46E1B67"
01026     "62E6F8E48B8FC4F4FF93EC60938B8E5C3719022C750C4309FC62ADA4E9028",  // PRN 19
01027     "D240216C5C4A70742CAA03AE910E8859C92E5A90A352CB8B45847BAC7793E1F75720D44919"
01028     "E896AD4581E1FD83986FF235C9834BEECAA1556794BE49033E79D4CCDB4DC67C5200E8B6A3"
01029     "EE891E700B348CBF092E4D3FA5E648B620E34E491D7B628A1FE7E2C45586B6577E50788687"
01030     "F0858C10F78F371B25C712ED2760C3D605D4ED4F052E88B66FC308D3ADD4A9B86F00CE4257E"
01031     "ED085EAE95FBB1E113FCB42CE12BB6076178A20903C55DA570EF8A25BA7AC8B7E134B8D4E3"
01032     "5AB172CA33CC97294A5E7E579B9361B92B49B63BB1982740015DFEC16882989C917F50D5FD"
01033     "D9166FE1001F3282D3C54A28AC7FD773CCC0634AF7CDF225F94107C169D2F2BB757EEB5593"
01034     "3CCE0FF116D7FFBA992F9A075A2439CCB369D5B5DE460CADC9F8C81D98E71651AEBFC2A918"
01035     "C551082D85F75675CDC8CCA1D3E486CFFB3B025D27C8D67C451FDFCF59C3BFA163EB791152"
01036     "390E9488C604B9B8116C329453A98F7A104527BC677411034CC49686108E569B7595E1DDC8"
01037     "5918D90BBCB337855860D6E4718C0679DAB6982D23FCB6648E8561F44BCF9B052D8B583845"
01038     "23BC592C9B7F824B96AD1A39AEBD2232D6D34DC171E8FBF933900960F207B55597759D23E1"
01039     "E7945075861144228A2FC100CC200D2B862DF3F26E6D1C9370373FE165C326D8C29FD2F0B30"
01040     "71AFD5215781BFB589F605263FF065B7A5CA3F6AA9DE3FD8BF5589BDE3526",  // PRN 20
01041     "8E7752C52805DD0A723D61F0BBE0122DF576A42B5AFDF9F196A766C9B3BFE296DC16A892FA"
01042     "ECEEDD8256D2B1AE6BFE5437D4A2691803043B59862B30D68E4FF94A0700D735CFE9672997"
01043     "24DA9D680200C898EED1C785E7B8CEB14F1DCDC73FC625F9678B4076035872220C2FDFE0A47"
01044     "E82ADF36C26F942797D608BA6B38A3AD1A967315E1F2D665B27D51E350F075531A179DB2EE"
01045     "D55547EA61761CD2B3962FCB347279117D1C7A7574B49FFE0991AF572A2B0C962A8A79800C"
01046     "FD524AAF9E6401C44569600F41F04422DB891D25B9F714713086BBFD0FB268E66A4FB10C0A"
01047     "BEEB31D0FBFBA20B0E4FFF404051596FC6F6C8093AD01807FA52041CD33007B205D15D47AF"
01048     "733966411A36F4C7B846D0BE049ADC21B89EA4CE0FBA414C005E66F36FACF3C43B474D47DA"
01049     "D78AC114D0171C031DFBE4A15FE1A22603CD79B6BB448B67A4DEDC97262F7B869C54F385F3"
01050     "682C744ED5AD6C0B6E16793920E6B45A024010896D5FECFA111CC9F0C34E728B32F2C4D45B"
01051     "8AA69B621AB9AC3D9D79B38BF205E8D0D19FAC44A76B9F5644526E06858F76B3EE2D74AEB1"
01052     "971D6B6E68B83773399AC32203164564B102B26C370A9FEC673C285AE0D1D3DF239D48B649"
01053     "2B89846EBED4618AEC940DC62AF4C3FF0D56FC9FBE23EE3B0A4890BA2665A88E9F40C4B6A7"
01054     "70F9630234ED10A3A7FF3C5BCCBA836F3EDC8B821AB18D4B1D51D9962C328",  // PRN 21
01055     "E682E9D8E92A7837823C9B7714D267F9CE290E9FA6CC0A8432D3F7507DAF6CF681246AA4C2"
01056     "323C6B53BCC6E53B31F49742EE5F4E6F79DC36727E98B06D0300ED21F0CF5F2B51D8304A51"
01057     "D0B498F4BFA39C0049B8117DAD334D4B2E37676EC42DFE0EED63B3726872CCF9A10223A8A4"
01058     "563BE8AC266E0697004921DCCEEA5DD80C62567FDEBF2AFDF030192831A6FD871F63D5DADA"
01059     "4B270AA9EC0ACE47E75BD19018CB809B548D4F2C24831C384DD2B807852F596BD4FE32CAB3"
01060     "A16899D0B100E9F96D06AACB8DA8D51DB0B0F600F3B614461F5238188B5EDA68EA753B6ACC"
01061     "58569E841BAF92CEE04E6E2626B1FBD01B9B67D1311B1C3D67427298E2D193F0647EA17D16"
01062     "FD7FD6A40A1BDBB320A1F5FC64B97759AF4EA92AAEB759B5DD30A726E9B8EAFA372FBD83CB"
01063     "FF0000CA75F219A95D6A3CDE38B8DFA9281609A20EE39B73FEBDF6A155359476D073E7153B"
01064     "C918C1191C9BAAF0E0F161384DAD8AFC31A3FC1E9EAFA495E22D18C05194EB85298AB0F042"
01065     "E447DD627904B73E6E505712DF010531C88E695F6510C78B443C731D7FDCD62EB7C4015AB5"
01066     "D530BD09CE5229FA4DC5642AF176C39D60FE070DF635CC5435136C7BB9C4DC83B0D382B9BB"
01067     "636A6C2B3838542904D53B862585FE6EC8960A9A77783D17B2D90506F5D60998602AE5430E"
01068     "86025C8864883CECD7CE51B49CC2953A2A41D7EF8027F1A83815BBEF6F6B2",  // PRN 22
01069     "F6BD4204243CBA14DAA15A256FBCD138B5D875E28BCC0BA36855E648434CD04F49935C3D07"
01070     "4DD5BA2EB82AB14E82C30991A1159E990D1D36DAF794853A23C499AB6B3DC02A89F0143103"
01071     "72813643F786BF19D3FA8C463EE50D9FA87107E91C461AD2E5DF2FC99630D2005894CB7698"
01072     "123111FAFC0C5BC9D1E8E84FCCA5179A6C9AFE3E369222D66854F90D2668A57FDEE00C300A"
01073     "EA4E88F03F05C4D7695B206DE9F7E1D429E5E6B65DFE05D4C861F4E7844DDB9062C0B6DB46"
01074     "B27AD0368992F54A44829DD11A05AB97BA8AD854E428B87F20C4E5E4BB1FF3803809A81F2E"
01075     "4C109572006729A5E490E0AA40BA55F4391C9FB758EFA79B97E6D413BCB02D33A00DA6705B"
01076     "FBADED66CFC21291C494B7C3293810012ECC61415E609DD97AAFFDEB795DE36026B4602DD5"
01077     "46A1AD937F1A6DEACD3393F5530C48A7974E2882CB327AE600C05A535BDE5D15AC52485958"
01078     "2EEE2D62194B73E01643359E7B2625F3EB9FE7137514ED549A3196FFCBC8072B4F6C18CC67"
01079     "AAFA0ED6029A805EF0987E2F27A3260F849C68F3EF91DAA9E579AA16FDA698CC18AE8706E2"
01080     "8C6D84CB3F593273D763C2969933D8EFA564E8C06C427809E6A5A6F76DE7C8B07FF4EDDF6C"
01081     "F2B7595066DFB15F5C6F3839DEE642FC86BC1F3AED7ED2E65B665198AA034817DBBBE0FE30"
01082     "E662B2161276CBD969FDA05AFD6D6A570C1E3CF7E324634441983F257E2BA",  // PRN 23
01083     "A9366308475F2D8D0C2D451C4A65A01EE58A0AF19B791D97382EC59A52616C7480B86EB1D0"
01084     "A83E93224B0DF73DE1D7EE6D51088F3B20B7937E6C0144E0DACA6324F0C8E5F9D93A8CBA10"
01085     "45E5B509D7DF98619FDDFDD7892C3082D69008D9D3ED6C9C1367D9DB7C04621D7CDD8A5A25"
01086     "99EE45B87A82F8CE8D60293E7A71D11700CA9AF117D630C5D8B876A9DCE519BD653114448C"
```

```
01087     "68B265813C608435B96CD642A420A15FBAB467692931BCA74F1F9D23F5BFDDC5B8651139B5"
01088     "A73F04FEF3DA64B7BD56E49235069EE5E8A136B921051F1D1C7D5993E6EEEEA2D58583152A"
01089     "DCD87AA89CF5962BC8341EF99CEB3682A2D0686602CE140ABC2FDF79A778A9D75AFFDBBA00"
01090     "C0BD6A8A8AFF9B5D1F30C8373572C81BD9594890102F46B5A393ED126C36AEF6A66E231A24"
01091     "6FDFCBD3DED198ABC54CF357ABC67AC83680C048932D7C902AB7DB16952B3C95DF4E845B46"
01092     "A362FFE1A27CD1388483FFA41AA563933371C0180848F9E3C03AFC1F00D6ABA29A953327A4"
01093     "E3D9FAD4616C8546C9AF89FB4D08D4256923B736A8F68FEA5A097E0640C16E0F7F942E6A6F"
01094     "5CBA76BB00D81C606C7FED908789A63F01F9B5FC7B7BE434E85A0A44B2070BE71AB2BA0132"
01095     "D9D7B32E2D2FE229619F85643E75B4141D355386D1A09F45738455BC21607086C7BBCD4B73"
01096     "F87DD83E905BCE8FC6C5BF1824E904C4F5C26518B2FEBF8EB06B22437270C",  // PRN 24
01097     "92D87BF3F54B0445C05E508E80F9CBC0502F0897D717CA232004362F394A023BFBFE3322C1"
01098     "D331AFC6454FC756FB48768693FD5C46DDB40DCBF14C726C24ED67D8F3EB613BA80B0E39CF"
01099     "0747DF62D258613640D881E085C377DE1C3D149C8359407C2C6ABC0D2718A2D42439A8E7B3"
01100     "8CD7DCED72AE750B2BE88D0069FBE94BD69A9A4B4AD42FEC5E651A31F86B90DC2FEBAA6FA6"
01101     "E5F6368B620C1750278DF393F7C5035D47897FC05FBC419A61330135F24365F13D653D77CA"
01102     "2930DBB05A3815FE83F75BB1BD8B2DE12A2FAADCD1ED62329C55B87FB32CC8F3B42D888981"
01103     "B4192480D1F57CEB0C55897BDA6B9C0ACE1E7E4595E30C7368306243208444FCF4574C47B0"
01104     "7725B25EC2E28F4C50B744B3860B361DDDD22D949AA94EBA4F97606FCAD91394B6FC0E634B"
01105     "D15E099E697403B2AE84CDF5DBDF36D91FB82C0BC12B984FEE83CA9E97C194CADF8382CECA"
01106     "AF49EB3BD446F660F94C188C074CC312E186BEE0F6585535B050C226659A94B4C4974DA32C"
01107     "DFF30DBEB4DEA588C6F490F7432DA5FA2408BBC931EAF60EADD7B891A61C157147B8DDE7A4"
01108     "5F909BD20D5B12009783DE410940245FE4E91ACCF72942E486AE773CD665912173EA29875A"
01109     "1722F8658C414CD08CBFDFE1DD356E167A9D7B20BF7441562EE816435A78BAE7E5A5EB4DA6"
01110     "AAAC36F594C93E2851D76B6A18B0B03B30CD38B97E38109C494C557643D58",  // PRN 25
01111     "BAA2716F115D72D2037841EF9138D19833C7C5FF40F058A960826E6903155777 10EFE64BB3"
01112     "7691564B3B0B6C577DA603CC3ACDFE1785541AAD23904758A5A13BDB018E7169D479A1FAA0"
01113     "31CA72FA6D6AE9613D6B2F82AB07500B49DF535F86A76350C140F9CD25295D6BC2F38C5D13"
01114     "C99540E2363862F06DDCC486D884999BCB840BCCAF2AB84F5906B9AA0F77D6432F65315583"
01115     "92641C52FEAF9D8ED86BF0158134129F34ECD0768BC02ED442254515A74999C6B8052A1FC7"
01116     "97F5720738C69DD9B3FFABDDC8515CD279B246EA7C67754920038C5A4C8D301119CEB95FAB"
01117     "2765DE39DDA84180CEBAAFBF4976118A8373FF6BBFC7FEBC3CFEAB1DA69DD3DB9E428C5949"
01118     "50FD51F4D98A393BAB96001461F2765834ED70C60BC56406CFCB3E784C59B91C19783E67CE"
01119     "6C86713C43DCDA9512B2E7173AFC2EF9A172C9CFDDD3000D7A981440AD994C39DAE6FC0B64"
01120     "5BA0FD49ECAA19E572ED0FAC748EC837A7D6F28A8D004402F71CA209BB9403B21E29836C5F"
01121     "E897268DE0736E985F9631DFDD1AC59D5411E684BE082F41108E33D2B92B2D45ED70FA52EA"
01122     "2D6DE121EB9F9C886DA479464A9DFD9970A406491E334372D7B78936095A7459BFAFF0E909"
01123     "0C2C6B6D62624A79334F879A5C92C685B50F75F04BA664EC95893FF40D62EEB24DCDD28872"
01124     "9D0C297DF5ABB83C77FC11D0EA3EF18E3BC7C2C065CAC51390C610B591D24",  // PRN 26
01125     "98CDFCBAD056240E180F347C00912F2D9ABEBCF5464D410BE6A50404B830F744D78F7D9718"
01126     "0404FB3BCCC2288B7991810B2562C4D509200CE1F9C4DF6DCA4C600D9ED49C9C1456141C7B"
01127     "71513E728D41970ACDB6C15B4A4E327B9A87ADA73D1D46EB0A21F2F5481C3B42931C51B780"
01128     "FA526C29B98E6B9C714B20049F7A05252CBB84B8E36026DB2379C9632A0843436ECB72D15E"
01129     "A2950ACDE18DBDC6DFB01BF08F7E191EC885F11D1D8B7BC96E9836B395108F6854545082A6"
01130     "94D5974CC36C8A658349186C1BA892DAA85D3F156BFBE94C73BCD815E7652C38E178AAF020"
01131     "14F0E6F23A4E7EF689EBF3ABDCCDD40E2DECED316F07E20716927C8F7B203D51D957EE6EAB"
01132     "062B99ACA0D28E0AB50B516CD92CBB9BA90333E73D58DE0B4B633D81EC93D15EBCCC813EE6"
01133     "3D63BD18517F4FE85C37469574B8122FB91388123E1D5E805166FB7157494F8559F90A4FA3"
01134     "DE9E71DA6FA7CCC6086E638BDD4FD3E4487506ACCF84F1E1678D714B86FAAD57A6B76E085C"
01135     "FAC30DE469BE32E2D203C63B43F073DD24F4A1E039B941E7A97F8BB28B516217455268B6EF"
01136     "BB0E1745C23D6D12A8CD13E5D242F562F56FE92496342000A731BF3DB0A7D3110705DFD0D8"
01137     "DEFB85665B77347CEFC8629F3757304F6129DA9845F6509FE3D32DE9FA86EA4FA9BF86FF7C"
01138     "C8E726C0FA9F93F889C467642C5E944501BEF8ED59793AF8804A9951B4B88",  // PRN 27
01139     "906F6C5A1D3BD03A03802EEF5937E214E87B5E2F0182BA2C258F44B516EC66EACB705E06EA"
01140     "6DFDB56600B8463A421DB03A51460091D7FE889E6DAE32EC19190E7211F08D37846CEE7364"
01141     "B6ECC07C1740CE990141C4DC4CB0AC9F25CAFCA6BC9111102EABA250ADFD505201FFF638B3"
01142     "1A77CCE7A1ECB273F9C8ED84EC2F403C1191596A53EAD823421EC47DC5E78F3BD1339532C9"
01143     "7E4EAA024CCC906EBFB870C1467C3D845A178EB07C11BE8D57E4EDEA7ADEF162923E952145"
01144     "1B871DF6E357DCEEA7F62022106F647DD8A23074AC10AA632C56DC32B34A4A184FACC64E5D"
01145     "1E8FD69269660543EEA2FD584117A3EBCF6268352F0212ABCE7CD28A93C9AF76722FB5A71F"
01146     "F9E5AC4579A2BA32B91818CDCB62C77A6A8EB1F4C34132EB463812B329B6B22108AC36E71F"
01147     "38338AE3A52C632796E45189632B73FDC0BD37A457204757261B7CFC01E06BC767A57A5FA7"
01148     "CFE43794F65398A94B4EF09D6DC2A8691BD0CB018BBE7B66E0C37BAA4723247AF3424BDE22"
01149     "614A9A581A7982E8C2323178BD2D46E6912A2FB2D2531819A180689D7F2C9B5C5AFC2DCF1C"
01150     "7FAEB1927EB79A72EB1203BB0FF17DAAF27D66022195890BBDDA786CF1C36ABFD96BC36FA1"
01151     "D2A5A0CC3D7EEE1A1050CA840209903CB9FF429C7EE9DF9CBC2BAB84CF28FAEF5BB45AE958"
01152     "8970A28B6BD9ADF8DF134C1FAB0DE274B5C7452C4836A573A26A0B4C14B74",  // PRN 28
01153     "C6D5046A5000ECDB54C872F2DC494F2DEB88430007C9BE8EC39FFB148F00F7861D8277589A"
01154     "C839AAD30AF7D7A2E0F9EE8217A39C521311E9BD59A71BC6663A7738669D6D3BB28124A80A"
01155     "BDF905DFE2C9539CCF0C8FA39EF84E9633D63BE0C32F3B2AA9FCDC18AC38C3C00924E9D549"
01156     "77BDAE61410F997038BE066DA6C945D8258B7DD133EECBA836A7A6A2907C431C522619D466"
01157     "430E6ACF15030F7FBA4F3D6BB545CAD85678E81898D2DE358CFF3951C8184066B18930DDA8"
01158     "67890871AF6F4133B492FC894DBE4AA5F1E44BD361C4560ABBCD3101B4AA4E065FD6030879"
01159     "5DDAEBADBB604A3D58776006CD074389AF49A0EF09586410015C7DE4FEFEEBEA6262B23571"
01160     "B93BEE15CDA2BBA60B6CC72A7DC9C80C81C9A25FE3D149C7A8BB2F704BE11177F92E2CEF0B"
01161     "BD12C0766D691CCF093D456AFEA411A8FE5F1C1A44F31017760F0D0CC3B271FB15F56D9F51"
01162     "A594C34FDFF8F8ADC91584ED8D7E1B6DAB27B2BE1BBDC4486FB1C822F23704BB2EF4B521E0"
01163     "2E42FDCABF69588B0B9D92AAA73116D26E8E9E48DE94F6267414AC845467597B4C1F2A9A8E"
01164     "1E82C0A1C05955022CDF87386098EAFC5BF1A040716A89BE53A36B143376927028A561BBC0"
01165     "7AFAF42494DF5BC0D95170D853DCCCCB22FD36B7947712EB369077D02BF85B0A4F57757ED8"
01166     "0B247E521AC640D1B1CE30F93DEBBE2389D364A8B7971A51AFA4F557A8E12",  // PRN 29
01167     "FDCF36E6C842D2ABCCE9D8783D0D7A7EB74992EACEEF6C618AC7DED4E457B1A708BE2C82B2"
01168     "8A9563F4A088FF7DB146B16B47A900DF49A4F3FA8EDAAFCA09F408B025D04EB673E105E0F5"
01169     "5959B7951CF0E999CDF68EA9B32333DBFE0516D272111CBBD9933CA8AD8AA6025E5F9A062D"
01170     "8305344CAFC3CA391BD8DEBDC58F7FDBC041B349900E397609C71E4EA3A9D8407C63E8A6BB"
01171     "EEAEFC92E9C939147920E48E35DAC6D123DA46E4F0838FD732E43FE4EEF6BD68D5AF0C9BA3"
01172     "A0CB28233743B291D4E10054F695DC10A847E661F39C4C133289B07ACA8B544EE3E2EC288C"
01173     "B18C40CD9A8E48A93378FD50E077EFBC21996424B539A397B3D2A6C7DE58112CF55E82E8FF"
```

```
01174     "10F75571A15DC248E6B77CBB91D8BF2D53E5C4E9A85C7EB8FB690F74BE029CE1B569EFACFC"
01175     "16872C5008820FC6A7D12AB43E08B4AFF57DE6B43B613DF8480ACA556E29D792C6C81CB1CB"
01176     "54A67245C571A04965267BA0F9CD3FA0950B9A5B393B4A230A41E455267CD396F42285F0E4"
01177     "9C5AFA0B53EC7B60C1C317EDA3FAE4B1713A80D4EBAD32FC685C13649C4806D6FD887A24A4"
01178     "F7AE801405EF28F058B37112A680F9E9AD0456314E9F490393CA25775797E4CCF9184FF0C6"
01179     "A237AFFA8DE1B84C420A6183B1D49D6F2AC1E673E7FDE161A8159DCEB00D85F032EE76E393"
01180     "1C459CE935DFE4AD6C6110591EE58496B82A16630E82320B9510880BE4E72",  // PRN 30
01181     "94964FC9F66389FE3880283C4250E6E19F195DFEBD2104FC0959E084308BC9CFDC6E5ED1C4"
01182     "B48B4ECAEB4FDE5F215FBED85A6CD4D1C1466E68A4CF21AEF29F77933549A3A6FF7ACD8AB6"
01183     "E6C689F1E8DF0AD8AB289D5C33023DF90B21A26320CE8C1CEB2C099FC1DB58737665855DCD"
01184     "20D587E176483E33EF14C80AA4760F751EE5B28460811E5110FEC3D689AE2A6E91D0A3F1E2"
01185     "2623E88571F4DAC895AA428D42634EC142E56D0D57CE68D7949BE13AF234229E546E9D66D5"
01186     "C58E510BF3EAC7B73309BE16DCE6E2280AA80247D9EDEDD20E06295C9876B412B786CF7E5F"
01187     "10737921581 31AFA002FE7750A17015A9C2580646A9A0D2A3F0243AF1AB4FEFB3D02850455"
01188     "3AF9C5C34D1A4A2FE3B8DD8BF8CEADA82AE63C319BD7981D97155AA2F105D724A8C09310D5"
01189     "C3168770621524 19A006ABF56AADED74DF0DF325D666C31DF51F194CFEB331E7DAF0041037"
01190     "2999D2D05B023B2C3067E6CE4A472FED3B8BE1C15C24DFBF4956A5B670FFCF128E5A230397"
01191     "64BE39CBE55636B83674060B3CCF5EF9A7B7EAB0813ADEE82E271C422FB78A9820007753B1"
01192     "E62BF4CCC074F7796D5B2008FE6542DC0C77ECA3810120ABE9F90BE5934E8EAE365D02B3D2"
01193     "DF4EA4A827E033263B113EEE5823DD3912FB31E3C4B46B274D7115F34CDA793DB6AD2CD8BC"
01194     "AF4B13B832AB605BE42B2877EE2E66B411668EA29A7DBA5BD969B9F152638",  // PRN 31
01195     "9B8071D96E7D361B2462CA93748DE4D31746972DAE582AD4F70A188CB40C2E6E418288B6A7"
01196     "13ED4B647013B3EC31C9EA6217DE55D016A1977A0B285224129CDC59A9E54F3E5094258F11"
01197     "C0C995F60785614E560764312CD86C6969B3274236EE602EFAE392C015E4C3972D6FA2A47A"
01198     "B48D5C5F6836AFA54F28CCC03BB4DAA0A1DC0DCA3FD3F2B15FB2ADD907D3BF7719D1D9A828"
01199     "4A47C30F32712A8CD440148B8DFDB851FFD25ECA2864150B832F8B5DC3A7C701371785A662"
01200     "85601E96D285FF88947804AA4D88665B3E15760CDE327FBD21393042BAF62FDCE6EC41955E"
01201     "877ECAC331D594ED40547AFED34D410714CE57FCB4F01C8826519ACB85F447306C86BD1BA8"
01202     "189E0621DD09451E8F341AE47E7FCF1FD2DE2AF78E0AFA27A4B6DD51A0710FC1FC4A599823"
01203     "4EDAA1D4CF0786B779F637EE1A72058774C1B4BF5E125DEBB4230645ECF87E3C6FDC91E1D1"
01204     "4397FA72686784815D9654839AE8FA43864709EE0F4A336E3C399CA20B2E652E2AB17719F9"
01205     "253F772EB7A9E8838FED4EBCD0F8CD977583BDCEEBBD925676F56AAB0C36F3DD915F6691A3"
01206     "0D60D523216FB233CBBEDE7FDFBA827450E595AB51237F9E77058E40F862D3A5A96E4AA3DA"
01207     "74503812EDEFA501E526DB6B4C642222D7F33B06D9CD0023471DF5730CF8E2BEE834D108A2"
01208     "5729C1C1484C207ECEB0E4598965EAB5216D4E7C30577A89FB8BEC0B118F4",  // PRN 32
01209     "E43093167F77AAA20EF9F4AB7D715CD0E86542A0523BDF0256BCCFB1FA987A6565F808C03C"
01210     "71E6343EF41D2ED4C4FDBBB1AB1D617BBBDB776F07B000922FCBB3718F5FF326A7BBF4BDF1"
01211     "48206411E86506FE5CEB8EC0F7AAD1DAE3BB46482F5E85D3B46DDF93F4BA19DF9DFD700C35"
01212     "DC038CF4E499A058995AF06BEE8535631966751A85571271E6877BE6216AFF16AA2B9046F1"
01213     "AB9126E1DCADBF40C4B390D23379D52C80C10CA05079D1F05CA294EAF7E2C66B884DDBB52D"
01214     "C85A45FFABB2BA6EB95869689486DDDE372DCCDFCB408496E647F33575BB99FF1699219D00"
01215     "D2608AEA1A47CF52DD21A3CEB3F043AAEB40EB27F042D276227C09CC8201304C55D254AC22"
01216     "8FF309FE8DB74C85430FCF9441EF1B51A1192703B140CE3703E5848477FB0BBC2E82DC3870"
01217     "2D4C6A78A41729B7EC86F9890EAD369DE4F25EAD61DB0B5B0ED764EC84FDAB904FAD9DBEDE"
01218     "BBE89F338DF52F7A6FAC8EA1D76001846802B328E8D1F110219BD6BB51214814026BFD6E32"
01219     "627B696C7990F53591CC811A9A09A1154A50585B882DE0B404D7B5846DBDC8C66E4AA0C7B9"
01220     "0B08CC96A5CF2352A6592BCA536359C863D639CDA60FD23326619A9014FD878EBFF3E73E44"
01221     "24B045CBD506ED744D3243725EF9491D6165D332FB12985022A71C7DD4E3F942AF5CDF2396"
01222     "F6C346088EAE89535E1B21DA8F7B2E1E12DC8A37738310248808B9973041E",  // PRN 33
01223     "F5B9B9204186466AF61609CC856B6B2B33D6F526EF1E4BE4DCAA1E1055F033072B3801B642"
01224     "700249C024414B3217D160513E5A804B3AE8425852CF5FB9F51D2601F9BFF9F5E4512A45ED"
01225     "8F615A6671315590DEC31720387533A4E090646F17244EB571DBD6E6A10890FF19809D935A"
01226     "5E88228D424FF2E54B1A37576C059A756290BAAEA5E36AC977526339CDFF915A763B2CD065"
01227     "7F3A3D385A5D92466A44980AD8A1B4E1593DC177E2447ADB53B1EAE15A91BD6703E46F01AE"
01228     "91D6C316C1DB983688576B3B2BAB280FF196D5BF545755097249C517E1271202E3856475A3"
01229     "C95324F827961DD7A827709DA2924B21949B6BF032675D80FD120180F6850BFC385E72399F"
01230     "840B80E5525EE70B89E1C524D580390D487D0CD31AF3CA5691067BDBE8E2E91E0099F59700"
01231     "CDFCFDF8B3ACBEBCB61DB23A28E80C3E463D9E53328C37EF8FF0D4FCBD287932C8197DF57E"
01232     "EBC9DDB29E2046C7FC14970F9F37CED37D8BB0958155192C8595E2A9369944DAF58FCC1D9E"
01233     "1987DC4FA77ACB256A65DA78B733178FED14DA3E3E41DB808849132EB05E48BCA03ED499A3"
01234     "14C857FF7C7C1FC8BD2478635396BD7F252ED984C102E23F55450F57ADC7EAC50A57B78BF4"
01235     "F988565A172E878638ADB1D992703D0BB742605EB76E3C85319F567D71B9F2E40251EEB936"
01236     "B323DE98496229B1F8A996AE4F6C0C2FBB3B2244F3BC57EB3FD067F535102",  // PRN 34
01237     "A7F21431EF8C75A1E9723B1FBB602EB4B92BE5FC8DE70183659A45A31EDDC46643ED1287A6"
01238     "9DC876634231D3AF95F2D02FC9C0C7BBBB12319CBDBFDF6CBF94E47A05204AB14A490C6FBA"
01239     "BCF329B5B0EFD8C2FA2884576451062439 8BFDED1B250003FBA61A67D2127550FF07A6FAE2"
01240     "CB5B8FEB23D289FA235BB4959F8845B25405A75A30E7D05D7BDEB721BFD4A998DAA6F1A547"
01241     "12901551E30D618DC80FADF7CB7863EADB7EEE708F1050AB0510A956C3C7706756B6CEAF61"
01242     "1F02656C3CACBFD9E4507AC2B4B78D0EB0BF757496C84C7260CA972F0F45A393A89AC2BD70"
01243     "F15E629473D61FFA29BA18A468D8D4B1F72DDFF99608AC9E6A6D6A9231F24388991E80FBE6"
01244     "A9269AD6D31182E79235A0E33CFEE9E3526FC068A9253789407AF8CAECE14BC337FCE02333"
01245     "B63224A61E913367CB36641A74E2DB2892C4E939A50127DBEFED7456F7E61454FA5016A6B7"
01246     "5E6133B0417C2860B2F640ED8EE62D5100EA172BBF8FB02E2022D29DF33C25654C96F8FB23"
01247     "05B72A92105E027444EE0F27769946A67E58D9A26AD01483FCA4BCDA4C00A8119E355CE733"
01248     "474349DC8311AE9853A064931FD68D598B02E0724A13FF63D88416FD5D03A962B5D8F79586"
01249     "E696C484D7F287773851831EBF33B0158F51C89C493C116157D20DE66F3990061135F5CBC4"
01250     "6862BCFDA5EB0322793BD6AE188698A2E1DEDDF71702B6CB01EC2BDF85680",  // PRN 35
01251     "FB2C4002173D1E1581890C14672C1F232E8A1332655108F24F78E53638EFECA3DAD18D19FB"
01252     "B1E91C6AA4761017DB37FA76F70BBBC4C8E8015384DC9E9EF14C6AE60DE55974D448DB8C36"
01253     "B030DD55DF5EC6F6C56DA8CB9BBAACF6784628B669913A7B291632730C11710D7E181401FB"
01254     "535FE8A9A0B81578CE54123A6C3897496375CD7C60BB4E05AD16BDD4CB8F652448E3020EDF"
01255     "4D51FF5EBB9FA93A740AE77941900299434835 8CD7827A63BABCD7C4501CF8ACD77A220005"
01256     "9DFF5B180AFFBF3CE52133C7451CD108AF9CC8FA32428AE9ABC55D89E1931E6607160BF960"
01257     "15901DC097B56299F9F358DBBE921FBD7D0DB43C9C5BF97468813C0AC632D35C858D8384C3"
01258     "2EBDEE933C9037EC36D2322F54FB0D3CF305801DE42B4EBCB5F83F3D3FCC1AF56910B7A547"
01259     "F50F851079B6D0D262676145C27DB362AF383037FFAD5E3B26BD2095143B361F8242573484"
01260     "2E2153C5F428D89DBA45EF59BD9DF1FE1BA941F172A685F08C18B76B6A8FDC86EED2867A57"
```

```
01261     "AF48E00467993864EF70D80CE5550C5EDF085E94435ECA89F6513D23A56235739534C777C1"
01262     "7BC6DC7BF6E04BDAA9B0C94C4D7AF1368C6B4B9D3C97D25A26B05703EBC4D1F8869C5DDA35"
01263     "9D189F4A3E548D53731FB429F3536880CAD9FE13F70DB0A080B949C3636611CBFF0F644ABA"
01264     "8B56D2C6E988B3392A7452A8C8E52F2288560331A78F4CF021D2A41B2ECB6",  // PRN 36
01265     "CB7A566070A06E759017D62CF78A429913370E6498BCDC038C1D3079A0A3E6FD4FDCC851DC"
01266     "BEA9EB3A7D266479F75C40452A282CB406E9A2581372BE2FFBAF6E7F3359EAA594631CEF87"
01267     "6FD62DEFBB3B816EC98A1F72B55F6023B572A51F320CD7ABF489AA905CE2274729C0EE0F67"
01268     "7A04D7449A489BD02704CA65BCD8B753B689CB90C87C73FB4A45219DCD817E9AB584EFF049"
01269     "EA711CD3E24CE41423FEE6F258A65CA8EC3A00D45371EC5B846CF1549CB5CA4FF14B696140"
01270     "D3483DDC7801BFE6CFEFB9189B5C7AC3DB9C0709F630D80F361442BBD22636B00530EEF21D"
01271     "96D4A12CCF85E0124939B025CF35A0A5F1B4C2FA5B5DFCB3772CBFEC1F0C1C824B5CC22BA8"
01272     "FB2DC92164FCBC60333EBF920A14599F1E8EFD4E8CB0257134947331D820595BF5630EDF00"
01273     "814C32994039B47E8784C29D0D1C5B5D219A9F55FC14CC18DACBE6A79698A8073276A424BA"
01274     "9A6606DFF369CCF5F9F4AEC9605F6E44807507B36A4918FDAC448E3134106683EBA80A2D97"
01275     "E12EC2BC1EBBCBADBDC631CE7D61BB9DDF143C3EEA4E36EAAEAFC001849812A7CDE3EBC194"
01276     "A51EA2A9A2AA15A0A4F29C2FFD8B2CB4ADEE99B59553BB2398273C04329A5C884989D417C1"
01277     "31D68E9B505594C9B9CF1C13AABF05093A9E4224D71B54C6F61F1255E25FF7527E47768FBD"
01278     "51523F42915133E02CD7E404147E7AAE95BE651BFBBD65E29130D0CA31C2A",  // PRN 37
01279     "D90A7D104DEF06F51CA5FE107C115C130B8823C423423608DEE00E0C2E6C202B60C5D20BCC"
01280     "9CDC18767D89E5E7857BABC9531C21B08436AC14C5B34EB986E5521369BC3D9B1FCD07C992"
01281     "40FD0DA306C69303B310B8F78D2BF97D6E9E88843234D73214431F6BAEFE5109AFDA22C4C3"
01282     "48A469370BBF9B0A81CF27ED919A0A48DA3E9061113C56659F7561175A608B7FB37A76EB9E"
01283     "5ABA5F4A6B485EA5D097482A28F4EF7DE785D31DDEBFE41DFF7C51CA2B64A0D2A7A70269FD"
01284     "E399FF7628202289EA97141F995FF51DB5AF7C9C5CA76F0AFB4981D021FEA9EE6A49ECCCD1"
01285     "8AF290B6C231265247DECE33B3E165FDE4FCF82EA84A4A8AE3FA90972D96DB68550AEF21FC"
01286     "E82E0FC75FCDE33C2B6F659A873AF09067D94B068DCEEDCA4D392BE5077587785CB6A1F194"
01287     "7841F483BE5080E85B49F1073D4778A39239266DB2A24B57B5F51B1F021D05C634D66DAD08"
01288     "1D1668EF0520424581E5E0B62ED085D005F169950F971E1493A49A60A595342AACBA236EB1"
01289     "6D5EF94E749889E50656D98F94EEF205445AC027387AE1363696185E24749EB67518B89057"
01290     "899548BABE29F3487D273C54EE2BE6254C7F3ED3D3B22FA3FC94791A7BC7C499A391179368"
01291     "D9B1F0DDC325F55022F5AB7B91142E2880BAC761D154C11F2C58F4A42E8B3DBCFD44777F08"
01292     "A74676DFB7039F24C9A4C4620726FE12849C9D3C15293A3DF7F4870DCCF56",  // PRN 38
01293     "9D238D97CE04FEC0D292181D145EA0EFA727DBB7BADDCC60357762D262DD7D7A3BB0189E05"
01294     "BE490018ABBB61B154F94A4FA81985E774019CCCE4B8ECE9858C0C2BDFFD7923B3354A2144"
01295     "ACCC5C9DA34E50299EBF68E9BA0A269760386E5531F8964F8024ACC99E7AC3F40BFA9D94EC"
01296     "A1B14A1B173A06867100C57883FE7333C079C5871A8A7603395A87387EFB92A77C5812EF54"
01297     "8CBC3B9178E874E45DFA8796BA6FECB5D81687EDC0AD5F804FF36A9CDDC58A5F991A2A0BA9"
01298     "99F86E663715CC79A439DEC44C394ECF1F14801D8F9F292A648D29D2E910032F224ECE51D0"
01299     "708E4ED6D3E65FC788A2C1D286750F4B6D768CBF1B6EDE8524C2A7457A82404A491DB5EC86"
01300     "8E6B95B02167A0C4E785290156943FA091F28234C4CAAB586F8095895FD21582E0356BC15E"
01301     "C7CEE893E70902CE4D01217474E0226FFF30CB78597726C74B95732E44D03A626F1D1DBA77"
01302     "85F55859B71C2DAFCF71D99EB99FFEF4877518C579D2623FF6EB326B8D6C7674826984CCAF"
01303     "98CE41724183952EAD69F05F44750F3B857EA01475555FD25A828BE1DBEF5E8B1C1237779E"
01304     "9E69D35E7BA9ADA124CBE710490280B64B2DC5EA2C41D6644012A18B369E3CA2FD93A254C4"
01305     "93506C54FA6CB4DFE24E0B5035230F2BDE6FF21101164F322B4B066ECBCA3BB4F3C476B41B"
01306     "D02D8948E555AF74EA13228F07A1BEDBF95FA7E16F2BDC5D178E74FE5A1A0",  // PRN 39
01307     "8DD8032BC8D741905EDAE449212146878D8FA15AB037E41C36D21B0064D65BEBB74A18F437"
01308     "0C80DB44EE93C2AB06DCC6C6FE7DEA83F270C9FB52CE00BDBF546348F0E6285A35A32A4751"
01309     "2823DD6E22FB7949071D4B58B46241544AB97B0F6023C70AFCFF0C7541B4B9E1FC6B3C3C55"
01310     "500251665D964871B9D1247B3A9728AE02FF47368053B1DE928B6A8D01ED3BCB18A3180EFC"
01311     "9700871F480959137CA1480C3640547497ED7CDB4912ADE1385F9F35B6D27BE64E362BD71E"
01312     "0981BE952B00AA20DC57465DA89B87BBE07F27454FB24B3B86766F95A5A5B4EAECA2620C3B"
01313     "F87C0654E6561398F27F96178EB1062E42F6E6AE0A669FC6C7170F217C850E822B4E70C903"
01314     "3A375BBCF6D62D11B3DAF7DA976D4A9D955B924E7F0A19CE77A53EDD424ADFDEE8558F0688"
01315     "7E82A936193AD5E508938DD3FBFE02DFEBD98C2272DEAAD8AD3B0FC86C3637A6CE694DD95E"
01316     "4FD55F433E151D35DD4C930177CA66322001110B9BD0B89A096C8C3A3431C100E39E3D2E65"
01317     "504A770F4EB2F19E9D8FCCAF15B8E58DFB52A0B88406A48A036193F3EA9F8ED2322FC69CEA"
01318     "EB9E2DD6AC627DCF4CB109EA05AB5DCEA5F92902F3E7BD457C240C958FCA7B17F2EDA12489"
01319     "61C9C827E99BE0A3D60B3E27E42B1FF696DD58E48C7B832D8FE95529B9C8F075E67294C1B1"
01320     "060EC6736DB73FEF7407B320D850B049D80F9F4D536361038C4BAEAD92F24",  // PRN 40
01321     "C10051089B11B578F56E6C24FAB973C5D1B4A2F937E4A402FFCC45267F4A3F4CF4EED3DFF5"
01322     "30838A570000F7E523D1B97FF798A3C6E12F9704126A9B6FDF606031E4D8E1AE39A470FE92"
01323     "9124FC95961B85A5A6E10A79CBA60F54F37D4DCCB13FA3B0A3A1729184C59036428532E3DD"
01324     "B3F7B8D373DC08842EBC2ABBFDEB5B73B2F9B7E83C81D5D0EB3037A32AC0E5B6CDCC7CD84B"
01325     "9289BE16106D1D019D47209AFD8DA78EFA0F9108377C187B170CD53F366B17899EB41903FA"
01326     "1627AA245C07EEA5D6DA78039B3C4C1D70DD28870A40EE3A048D8209C4088A0847D6859513"
01327     "E01E0F4172C383D64F0B307D721C01495070336A753A75E1CA433FA973BB131E558FACC298"
01328     "C0A6E934E9F717E19EBCEA11EDF3736D4EAF593A8E5AC8EDFFE4D2B17B613E024B41AAD19B"
01329     "C0A27A1E9BE9D25344D4350596B9B0A48AA014191511B26E61A9753D8A38C4E3BECAC93E67"
01330     "CAB5696888C06B9099E9AE856486436FDD1AF1C193057AD1DFBBB14AF772F7D9B00F37A6DC"
01331     "9E805A8DFD9D53A161885E5C2C2A6DED0C54D00A8E0704877DA2F1EB6F6D8EB93B5604A4E2"
01332     "545D454C3D4F0D606AC192E170D44E85E2ADC91E54F56906909024DEDB6F9050BF3D3928D9"
01333     "AE23D9C42576AE4DAD1336F47519EDE436C3B1829CD2C5A0AF63476C7928B32F5E51826D84"
01334     "4A1DF765C5578F9C2E5F79024BFDA96FB3199A3C7B47D3755FAF376761B4E",  // PRN 41
01335     "9C6674915677A62A7AFD4FCB27B45F8E6DDDD08980B7DBE1131E6E2425C5ADBEE8474E93A8"
01336     "A2D4E495B195A6B83F2CF6472A36E690928362B9A1FA994A9FEB7BCBA7FDCB771F59FB434E"
01337     "D289CCEABDAFAE29113389EB98D6F17E5508D5976E11BC8A1E93AF9F7B1C81686265930B4D"
01338     "334568E3F29E1C2F58A62572A610016C1C1C1C9E1D0EB3FEB2B3A210C59EB3980C44BC656F"
01339     "A7C5E05A4472D4255B40B8A1604FE39D8B5026A976310648D5C84CEBC87A8BF6545DC843A3"
01340     "A0B64DC4CCAF2D2203122DDD75BA42E096844899A35A899FDFD72C26E3392EA03351DC78BB"
01341     "9F62F51D913F8008DB00969C64003773FB2014FAF97E794A45792495BD52D7BE7ACA47FF2B"
01342     "F570CF88303377092B5B6BFF3B01D38A53E8A68B0D81FC2D1D375EB27C7AEEDD70679E8DDB"
01343     "A6DE656442ED951478FB96A979B4A9091F344EF39AF23DAA886C6FBAA8611C61686332C630"
01344     "690109E2869D18EE7A2C21B22921B9E3DE40BF063E370FF64E7AFE160B7EBFC4AF6AEDA043"
01345     "042552F5F36C2CABD339FE1442242EAD931D1B83968D1A31A7E32A0838401DBB9C1034D56A"
01346     "DCAF5942462EF63440FD70F91520137A50372D0D125A6285F7D715FD9225D03A109E1FC5EB"
01347     "547303CD7708F88FEED2814607171930436B249924714E8D8E024C24B3C0C9E40127DECE1A"
```

```
01348     "D966C3F9DF01793864615F291B73F73D27B624ACBEAD3D371B8D4FDA823C0",  // PRN 42
01349     "A1B9E8AB6C44A3610768CC17E3B899CEDDB44B746319EC50BA7E006EA0F3C09E9D67EF8A20"
01350     "154B2F93EC265B800503BC72831800CE0C9FB6CCADC8327A840F69DDF29BE212BEFE4AB181"
01351     "0026FF786D9D3A88B29745EB61131FB47B385F83CA211B4A2BC0449F7ECE43B2DDAC94C109"
01352     "55FF2E078AF573FFA7349907571AF501FA6A2FCF24E4B5676F8213475351C0398748FBD60A"
01353     "5EE8A1128C0998E57D076A201453EA7C70B1F171E7D084F444311C9829CAD03E0C2330D7E8"
01354     "D39DB6B7901C30787FF123CFBAD4E974A5F6412FB1BC927FD65CAD1F3AA4ED52E2D84AF625"
01355     "7EF311222467DE179E89438524116D8E0A4A94C19A4D66D0AE0FC535D69C4953C2E43136AC"
01356     "07D6A5774D59DEF1471B4097146C2124DF83D36678FBD7BEA6E258D826645DD88AC9A7EA5E"
01357     "05FB49F2CB29EA9081323FFB2A00F3915D1F36675BA1C9AAA3B166D9FB2529150D3573E502"
01358     "668E33DFA5BE95AA6C2F1F106D69218839BA590406B4FE4A03C4B16E29E6BC31335234BCA5"
01359     "5D34955880B702F08203198EB7BAE381B231FAFE513F24C8BA9E58798F35465C2679DD5BA8"
01360     "B16EEDAFB04E5A807B9BE6EE4B3AD77609D2B9E0985BB23397BBDE08F4D0B21494A3129857"
01361     "E37D13862C1FF77A57AC663C206A92F8358E6C05FDE1A9D68131BC04B76F6865E8DF5C48F9"
01362     "424CD9AB6FEFB3B49BA8D32C923CDB8602C5367AA9ECAE48AEE6A276CF25C",  // PRN 43
01363     "ED1196222AA470CB4FC8A0B7D2C785EB45D232E06453B77DB76F60E5F7BAA0FAE80467EC8C"
01364     "656FC27FFDE8BFAADD2368B41AFF69460495923942ACF85C09EF01BCED0076960E5CDB1D36"
01365     "061933C1B6037548D27932CDF969FD5A910D5564F3A6BAF896A2A69F40EDA76E813AA26153"
01366     "0F686042235A39A76081F58D952662894E6E648BA72098D840235EC4A2B963F8E74B52F2D0"
01367     "088DACA7EDF48115618DBBABC698C04AAAC0632475BEE65BB26A4EAA08F0209B75EB259B82"
01368     "6724F580209F8B991808411F823819CC5F42A9FC1F0CEDBDC54C046EB1D1658FFF7252908B"
01369     "09F8C1D82E5F6D605F0B17B25506FAF91D4B3D4EBCB7A705E5BCB2A6FB409ABE4B1B0B4763"
01370     "0F1AB0B40A3455181678573F82A7C139F072659D80671961D614D9F7DA72914BD5D6F417EB"
01371     "C152595972FDFAE876176923CF1F2F745A6FC27E134CEBC2D9AD5CC733F619C1B296026B7C"
01372     "4717D2086D91765D83AB76605050B810C542819BBB7CD87EA2305AB4A567B2C500D9913B7C"
01373     "6046F748DEECFE3C3AF829EF97E03799EA50F7C9EB38990F0D466498C091C3ABAA18453B17"
01374     "3F0AA3A486DFAFEEE68E7998EA30E5C1349F2A54AE753963CEF58332A114F964B83B4E446A"
01375     "44147B11FFC60DA25B5FA705F34725B1388B17E99018236BD646DC714F8B7CD3BF2648063B"
01376     "2D3FB713AB94D82ED81D571D5BB4D3E5F2F067E2F2C90E8B6857E6A651B8E",  // PRN 44
01377     "EBE597534C0A6FEE9CE6AD10A80D2DFD160D7DE7A20154F62AFD7C3187D51D09DA39A95843"
01378     "2CDCCBD83FCE19131E65A30DE99DB796D93D0944223D960BA233798323D368DDBD407D51B8"
01379     "0350976951AE0E50CF409279301939624C741D20D5CDBF4BC3ABC8723E267BE75707024F63"
01380     "98CC6D32A7BCD0989555E0EE49C1A2354D1E961CA9910A5169ADFA71D56745181E0CF13469"
01381     "D8F3C6020637B8118ED8421A680FD4D515DF6B31C39ACFA36B601027AFBC861493D34ACEAC"
01382     "FB501DF9A311B7EB9D38D7107E6B5ECBE74FD35CC8BC8319DF9F1948653C94C868DC24AB8D"
01383     "7E650D8F70A3D1D878558AC9BFA50D425604037F3DA323192654F1296C8320AF934AF60DAC"
01384     "AD7F9630EA4ED0A635678213F6D09D73A2544C779596990DA42079C23581E22F541E6A3B6D"
01385     "3FA383597A8BDD7CA42930816D74FE8B12F92CBB9303CA5FA12BF5FE954FA7B53E0D88B79B"
01386     "E263B4B55B42A7672F5C4F7E6E2DB4C02AFD6034B7371E01215CED3F73AB8E3419D7447781"
01387     "AE5DEB4F4547C5549E9B0AD59940C162790650344E5ACAB33A187A58E29C9032963F088C04"
01388     "4ABE73A855FF1FC93AB8DC258E419777311B7997EA95D32CF1F0140CAB9E6707E73E4FE575"
01389     "E0752816122C39B8D12F34ED6771EAEE192D830797DBE97A0BFD00E39CA382D44F080BB4F8"
01390     "FF6FAAFF2A82744C6E14D3036E00EC2ED5C9D184ECCA300030D7BD77B27DA",  // PRN 45
01391     "A56990B45E9AA5F9523D563D53E854A47629A4183E85CA1995BB374EE5601BDBCE5C740799"
01392     "9D28FF0F1B3EC7A463D3C941C57B7EB71E6262DA3A7136627814EA6D28C8D16E91680B0D96"
01393     "1AAFACE2CA4099F986CD45A2AB387E39497B13678447480514B4008F0FA3B95AD7D2B4751C"
01394     "680F0C7D3651F9621E1355CEC76F10468ED33DE4C7D7D68B786B7F0006C604BCF893739FFE"
01395     "6AE56AD5AAD07D5BE76DE1CD581B10EDFEC9DF65AF925C4C1289837BE1854A62FCCEECCC4E"
01396     "2AB47B4864C467F5E4C276047501EABE444E051DEA933863E0FCB2356CC0C3817BA59B2E0A"
01397     "D78BBFEF4C3E4C35343331CD81C7F4C428B0418DFA4D6FF655BC50AF49C9A5E8A67BC265A3"
01398     "F30800B93E42C0E66F117E6C59C85BA5E273234EC7212C7B7AC3D87FC0E0EC5FBAE6654461"
01399     "F24BBDF9A063FD0E3F98741C5622ACC8BC008EB0E3F3D57680F12E52FB94CD957EFDD49BF5"
01400     "12D9131757161A73DD9650E3561FB31509A908D3EA8DBD1C3AA95AC32E6960E0D17A218A25"
01401     "784332A88E85303DEBE117BEFEDF46882995AEBD704AEAE032668AEC6B8E5BD28D3F110C9F"
01402     "8C1DCA8FACFDC1188A073037796555DC4FAA26ACB9F51B545D32C224424BAC3DAB20D3C08D"
01403     "784E9FEDB47547C25671C269A62163742B55AB4CF9406D2C5047F92C76356F0B90221E019B"
01404     "31890C0B8FCFD04E6B5D06114B967D1358797238D5B14D824C8F0BD7B8BD0",  // PRN 46
01405     "D85AAA53F2DE2946EB0CA09DEBB6CB61D91D27C8907B90C89E20C01F681D33BF0DC70B6C79"
01406     "F19E4DB64793505A0E055C33D1E07B7F5AE09F7EFF0C8CEBE80C84373804A48A945CA406D3"
01407     "A1B17CA787DA265C1D8FC735CB098C1AB37A0452F1A287B8E19E3A57E59EF8F2E6CCE2F0AA"
01408     "7772955977982B1F6880AE8717520753636591BA36E351310E98ABB77E26E2CA5BF267935F"
01409     "C38DE173CA20C3A964B56924A5E82A9E36D005EDE6680D543C021C7A9F53DB69BFA3201880"
01410     "B7797E90D31011AFA17BAE836505F5B337C23A07D6AF167258C314C3001291DD9FC565F6B7"
01411     "FB74D99C1B42B496422B3D3E1A564769C2306E40264641EC14724301795B4D54F5FF8D6C4A"
01412     "B3645B56C1D0FFE9B977FC5017E2A53392E7458BAA308C343AAFF5B46B808BDA1FE5C28429"
01413     "9A9EEEFC755132F9D3B475E2FCD6BA84917C601B5E8340B5A68148A33563E40AC5A1E49BD8"
01414     "D5FA77BEB6B99E6F4F68A10ECA112475531E7C1942AB9504B8F76EA7BCCAE4A19EF859FD6B"
01415     "6753F07BDD9A19421A2A0B8F27643B2311FAD7978F4366C41BD9F609486FC6EC7CE833A00D"
01416     "7A48AD816412BD201F40C93D136ABB8E329DDB816D1F59A1F3124D6314C64C9B640E4D5BB2"
01417     "01CDB4490A65C73228E78055AD6632D81709AA443EBF29A4607A504DD5CBC0D271FE595677"
01418     "B8E6A5E29E23D171D50B878D0D3F9416657E3997D077677B87399A6D1936E",  // PRN 47
01419     "A0E629454DE594F7693888C9ADB907F83EC6818D6FC0019B70077BF1B0E3C2CB16A263BE14"
01420     "5EA8129FED6BD98F3D9120BAD28379EC8E36DEC69C59B37BD6C2B65D172C70C69C23E29A61"
01421     "43121AEB9E6A9AB46C3DBC3979BC876F4357F6BAA5A0CB33C3AFF1654FE547D5D760E8CE09"
01422     "5A9B324E15CEC1596BB7E0D56433D8908EA69925A5CCDCF01702811AD1103F237D36C5F12A"
01423     "3CD938A61EF76A3387340499E4EF1855F892F57AC39AB419F1A397CD43FE7F4C5E609ADF3A"
01424     "2A8E401FFAC870985826CA0DE211996A3549A203B798A5162E98FFC8B5CC2CFA68D618AE80"
01425     "D399D8DFE0FE4AA81ED9B586EA89F646D5C0AA8B107506C7F86879419D0F717E8F70C5A9DC"
01426     "2C3F122CB52F3B691087EF5971C27086512F1890494ABA38B1619969F8846B8876B95C5614"
01427     "2F5B171502FA44ACD0299E82FC94EBCD4CDDBE1F5DF978871128F79C1758BB512D5C3BDE59"
01428     "A89786CF4E4716F0515F45A1CA31B06B23E4883F99EE0488848EB0A60FA14AABBD41EC5421"
01429     "D0C162590FEFFEBB4DD0300221471DE8DC0C87DB3F1D54DB11744F8A7115DCA0595F6CD99B"
01430     "74A6FB10C0D14E430F8B40E3E7105C89B2C8F12B37A8879AD9BEAD49AFA1BD56569713CCDA"
01431     "3FB634C3011D51A8DCF84437251A7B19EB61FCE5DF6E2278784722BABE384BA5424646FA23"
01432     "F6D10CE23DC5E0D8D512E7C5C094B1627D6285B57EDF38A7DB1BD57B72044",  // PRN 48
01433     "CD7AAC98501F29507EA4E0183E8A40D2E5117E47BB5D18D01A3732DE4C821DFE86521CBEA7"
01434     "DB29BE1148BD544ECC681689BCD1B41EAF755310B7659342F8EE11CB41550CC30E566E1927"
```

```
01435     "96B66C1A83C0B28BACCFA6C393043A0A2CB89712BC1CCB174DE58E66896AF39C1CEED1E05B"
01436     "0435F8CF6FD920D100F51584FE24879987399481DBF27DDB6286B6353919E552E669290CE0"
01437     "2AB4CD5113D7F484229F379C7332767EC69E4336439B05DE1C1E3563DD303A4F580BFF20A4"
01438     "0E49CB0822F715ED0221EBCDB5DBAD751124B1715E82F37488265135B6C8BBCF4F801ECC4D"
01439     "3525FF189493AD4EFF0C042B070C4CA8FB1FDF43D79F06A6E4E3D35D7B07D4B728D5DC54EE"
01440     "DACBBBA1EDDCDC07ADF7DFCFEF835E44DF1FF66DAF2A7BAEBE218AC3B15E183044D6A8A89B"
01441     "3C101B40BED97ED5DF93BBC1B84931D56B8C822A6D058AC74CFA4C85D8B456698E82D5B757"
01442     "4C17B041E5F4BEED09F75012355CBC322B822C63F10C18A8F279E9A0E18E1FEF183D23E138"
01443     "94E31F6D046956FE8A647558228F6D4D6910151EC03937876B6ED7A078D33DAEB3F2239353"
01444     "BB8181E62B286BBC41588DE10F478A5CE5B508F205A41820356767B0A0ED4B8DB9EFE34836"
01445     "2E9A90D6C30218B295B338B51C09239D02FC8A1E7DAAAB60AC37F5E67CFC88EEF69567B5C8"
01446     "1A03B449F4ED38B9D295A36AA3503173F6F6F66D93CE72D753076040FACDE",  // PRN 49
01447     "ADDCEDB50E907D20E826E6E8A0D30C20C74B2DF204EA784BAE9F618CAE33A3C937729DF9CB"
01448     "10BA2A4C33E0182A37200C0CC509729D828B8A2A20F283AC4F9306596684EA3FB5492A4C9F"
01449     "2DB459E7531C9F9C0950E7D9E93B3EE5912AE7E39AC8F4EC14B18F24E325003F477E347C5A"
01450     "C1B67CDB11AF3BBBBCD0AC3703024B5767AA67A208254F798684BFD1D3EACD757EEC772549"
01451     "50A146620400DB95E694574F739A991EBA771EBBDFF1056BB39A77DBE0636A032E17141332"
01452     "F951C57C6C90F348F165E3ABDD60D429D5D6BEC7E3E3463806F819EB2D212B3528A5EDE51F"
01453     "235AD100A35E890955F8A1DC51FDCB53EABCA2540997DD054C1F5B29462995B8876B44D0859"
01454     "04E55E1B838BEF600A992EB49CE078DF75AF3D0F137685AC0D07F0BE1EB87B63A41E74DDE8"
01455     "69C8A683BDE60AF5D77FF18F7137495BCEFD0ED28F62F9C3E25D332B5F861D999FCDC0B485"
01456     "1A984A4DBB53401FD40351ADA4335C702BCC8D900C737507B990BDDBE91D201E3A0946DC96"
01457     "8D43FD10D04B0B76667FF5B4291C2124B0124C6B710A6D1BCFAEB016B9DEEB0F7A4FE044CA"
01458     "4EA0CCD84B7682617C3A545071EC295B0663B3F577D562DE1D9DD80DE6A1EFD6D5991EB524"
01459     "6F1597B86D0E9A90CF6DB0EB2B8E7BAE9431E567F01AA98502C773742246467ABF911A91A5"
01460     "1F6C1B9E0C3233DC1A37D17DB91A5F0F661B0EB5886964456C7818601BD0C"  // PRN 50
01461 };
01462
01463
01464 /** \} */
01465 /** \} */
01466 #endif  // GNSS_SDR_GALILEO_E1_H
```

13.696 Galileo_E5a.h File Reference

Defines system parameters for Galileo E5a signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstddef>
#include <cstdint>
```

Variables

- constexpr double GALILEO_E5A_FREQ_HZ = FREQ5

  Galileo E5a carrier frequency [Hz].
- constexpr double GALILEO_E5A_CODE_CHIP_RATE_CPS = 1.023e7

  Galileo E5a code rate [chips/s].
- constexpr double GALILEO_E5A_I_TIERED_CODE_PERIOD_S = 0.020

  Galileo E5a-I tiered code period [s].
- constexpr double GALILEO_E5A_Q_TIERED_CODE_PERIOD_S = 0.100

  Galileo E5a-Q tiered code period [s].
- constexpr double GALILEO_E5A_CODE_PERIOD_S = 0.001

  Galileo E5a primary code period [s].
- constexpr int32_t GALILEO_E5A_CODE_LENGTH_CHIPS = 10230

  Galileo E5a primary code length [chips].
- constexpr int32_t GALILEO_E5A_I_SECONDARY_CODE_LENGTH = 20

  Galileo E5a-I secondary code length [chips].
- constexpr int32_t GALILEO_E5A_Q_SECONDARY_CODE_LENGTH = 100

  Galileo E5a-Q secondary code length [chips].
- constexpr int32_t GALILEO_E5A_CODE_PERIOD_MS = 1

  Galileo E5a primary code period [ms].
- constexpr int32_t GALILEO_E5A_SYMBOL_RATE_BPS = 50

  Galileo E5a symbol rate [bits/second].
- constexpr int32_t GALILEO_E5A_NUMBER_OF_CODES = 50

- constexpr int32_t GALILEO_E5A_HISTORY_DEEP = 20
- constexpr int32_t GALILEO_E5A_CRC_ERROR_LIMIT = 6
- constexpr uint32_t GALILEO_E5A_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GALILEO_FNAV_PREAMBLE_LENGTH_BITS = 12
- constexpr int32_t GALILEO_FNAV_CODES_PER_SYMBOL = 20
- constexpr int32_t GALILEO_FNAV_CODES_PER_PREAMBLE = 240
- constexpr int32_t GALILEO_FNAV_SYMBOLS_PER_PAGE = 500
- constexpr int32_t GALILEO_FNAV_SECONDS_PER_PAGE = 10
- constexpr int32_t GALILEO_FNAV_CODES_PER_PAGE = 10000
- constexpr int32_t GALILEO_FNAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_FNAV_INTERLEAVER_COLS = 61
- constexpr int32_t GALILEO_FNAV_PAGE_TYPE_BITS = 6
- constexpr int32_t GALILEO_FNAV_DATA_FRAME_BITS = 214
- constexpr int32_t GALILEO_FNAV_DATA_FRAME_BYTES = 27
- constexpr char GALILEO_FNAV_PREAMBLE [13] = "101101110000"
- constexpr size_t GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5A_I_PRIMARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5A_Q_PRIMARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][2559]
- constexpr char GALILEO_E5A_I_SECONDARY_CODE [ ] = "10000100001011101001"
- constexpr size_t GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH = 100
- constexpr char GALILEO_E5A_Q_SECONDARY_CODE [GALILEO_E5A_NUMBER_OF_CODES][101]

13.696.1 Detailed Description

Defines system parameters for Galileo E5a signal and NAV data.

Author

Marc Sales, 2014. marcsales92@gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_E5a.h.

13.697 Galileo_E5a.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_E5a.h
00003  * \brief  Defines system parameters for Galileo E5a signal and NAV data
00004  * \author Marc Sales, 2014. marcsales92@gmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_GALILEO_E5A_H
00018 #define GNSS_SDR_GALILEO_E5A_H
00019
00020 #include "MATH_CONSTANTS.h"
00021 #include "gnss_frequencies.h"
00022 #include <cstddef>  // for size_t
00023 #include <cstdint>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
```

```
00028  * \{ */
00029
00030
00031 // Carrier and code frequencies
00032 constexpr double GALILEO_E5A_FREQ_HZ = FREQ5;                 //!< Galileo E5a carrier frequency [Hz]
00033 constexpr double GALILEO_E5A_CODE_CHIP_RATE_CPS = 1.023e7;    //!< Galileo E5a code rate [chips/s]
00034 constexpr double GALILEO_E5A_I_TIERED_CODE_PERIOD_S = 0.020;  //!< Galileo E5a-I tiered code period
      [s]
00035 constexpr double GALILEO_E5A_Q_TIERED_CODE_PERIOD_S = 0.100;  //!< Galileo E5a-Q tiered code period
      [s]
00036 constexpr double GALILEO_E5A_CODE_PERIOD_S = 0.001;           //!< Galileo E5a primary code period [s]
00037 constexpr int32_t GALILEO_E5A_CODE_LENGTH_CHIPS = 10230;      //!< Galileo E5a primary code length
      [chips]
00038 constexpr int32_t GALILEO_E5A_I_SECONDARY_CODE_LENGTH = 20;   //!< Galileo E5a-I secondary code length
      [chips]
00039 constexpr int32_t GALILEO_E5A_Q_SECONDARY_CODE_LENGTH = 100;  //!< Galileo E5a-Q secondary code length
      [chips]
00040 constexpr int32_t GALILEO_E5A_CODE_PERIOD_MS = 1;             //!< Galileo E5a primary code period
      [ms]
00041 constexpr int32_t GALILEO_E5A_SYMBOL_RATE_BPS = 50;          //!< Galileo E5a symbol rate
      [bits/second]
00042 constexpr int32_t GALILEO_E5A_NUMBER_OF_CODES = 50;
00043
00044
00045 // OBSERVABLE HISTORY DEEP FOR INTERPOLATION AND CRC ERROR LIMIT
00046 constexpr int32_t GALILEO_E5A_HISTORY_DEEP = 20;
00047 constexpr int32_t GALILEO_E5A_CRC_ERROR_LIMIT = 6;
00048
00049 // optimum parameters
00050 constexpr uint32_t GALILEO_E5A_OPT_ACQ_FS_SPS = 10000000;  //!< Sampling frequency that maximizes the
      acquisition SNR while using a non-multiple of chip rate
00051
00052 // F/NAV message structure
00053
00054 constexpr int32_t GALILEO_FNAV_PREAMBLE_LENGTH_BITS = 12;
00055 constexpr int32_t GALILEO_FNAV_CODES_PER_SYMBOL = 20;     // (chip rate/ code length)/telemetry bps
00056 constexpr int32_t GALILEO_FNAV_CODES_PER_PREAMBLE = 240;  // bits preamble * codes/symbol
00057 constexpr int32_t GALILEO_FNAV_SYMBOLS_PER_PAGE = 500;    // Total symbols per page including
      preamble. See Galileo ICD 4.2.2
00058 constexpr int32_t GALILEO_FNAV_SECONDS_PER_PAGE = 10;
00059 constexpr int32_t GALILEO_FNAV_CODES_PER_PAGE = 10000;  // symbols * codes/symbol, where code stands
      for primary code
00060
00061 constexpr int32_t GALILEO_FNAV_INTERLEAVER_ROWS = 8;
00062 constexpr int32_t GALILEO_FNAV_INTERLEAVER_COLS = 61;
00063 constexpr int32_t GALILEO_FNAV_PAGE_TYPE_BITS = 6;
00064
00065 constexpr int32_t GALILEO_FNAV_DATA_FRAME_BITS = 214;
00066 constexpr int32_t GALILEO_FNAV_DATA_FRAME_BYTES = 27;
00067
00068 constexpr char GALILEO_FNAV_PREAMBLE[13] = "101101110000";
00069
00070 // Galileo E5a-I primary codes
00071 constexpr size_t GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH = 2558;
00072 constexpr char GALILEO_E5A_I_PRIMARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][2559] = {
00073     "3CEA9DA7B07B13A6CC0AE53DAD1EE2A0FCC70009338C08AC0EE457F76A1690815C3C940AB7"
00074     "22487CC8F3D1F4C428828E7FD2A21230E42A3BBDF1E792165F644D0E0335F95EBDC93D6005"
00075     "CC0C680DB7B0E1B8C4946B7974319F9816141DB9E01011E4F20DA8F1B8E15A6F618CF599C3"
00076     "F5C1A1B276D51318ED4119BCE0ACD0332F3DD8F88EC5215AB311C51FF4987DA93B09A43BA8"
00077     "4CF08032F6CB28F43043C54586811D870AD6FA27AA63785345C8BCDD3DA26A0134738BC7E0"
00078     "8461D5409FF0B791D8574CE797FC5EF7821055028CB4AF92AE1088F8806CD55F0E5FDFCD8D"
00079     "74ED801B2B44AD5D79D1924D41DDC6AB2070B5360CB64CCF487FE5174203480CC39BF50BDF7"
00080     "8BE7DA91542FEAB689457B3EE69E43C75FADC303F31032FD96B7DC70A88C3B7BAC7322B285"
00081     "D9CFB3A93AC8B890165F23848FAD8477DBDD3D0AA4CB3CD73A48000B6D134DA2DA70B56E59"
00082     "0A101AEE78864DA0C64A7BCC6B37CD6F31E9AFF10CA4D47630752D253944632DF6EC60AECD"
00083     "CD223F29399CDA3B74D1DFA5471277EE6C814464A8C55D3C0B83B36B6AC9FA90CE876ACDF6"
00084     "5E3EA3FD61D309EB71ED29A3D510B2F4C0B6D6C5B57EC9060CFBE48389DCB17CBB2284E7F5"
00085     "78565B91503B06F49CF3E8534870AEB6AD9707265A9A1E6E2E5E6DF6DAA367239A96EF5B02"
00086     "C19A4543D537EB4D9D73966C09E9B52B4706F57B3E0987885EB84DEA26F7823D895F620151"
00087     "88ED38C04CC6714F797FDB0BC713E3D0208462F9A68E3872A167BF1BF9791AEE8BB73CF527"
00088     "C50975B55C4E5C2F2E95B677F833ECC878D1764839608CC1108A75EE9E58FFCFE4CB52884E"
00089     "7AF15EE0632E0729DA1CF5B7A227028CFE1E08F8B881E1A743D52DD27BED33DE0EE75DC031"
00090     "B4864CF192DFEAF64F726D73321363A233F81C57232432D2B0A5A4C44F4320847A9C143F37"
00091     "8F204185D2B571482FE45D6BCA152E6EA7223BFC6DCE06CEF90CE9114623EAB9B1EC789B20"
00092     "51B4AB711DABF5B16FCD970F437B8860313B4F1F14D384EE3976B7E55D2FDCB7E1BD9BE18B"
00093     "722E37C853ADC7E1CC2870A02881F95B78487780E1D1C296415109CF07AB63D0782A9F451C"
00094     "BEB3E8B919917AEDBCA8A8E563AD3784639793E0F25CC9CC62240FA04B2F141E71BF5C84EA"
00095     "C56431159556B8BCE077A51469A87737D3D6F06D97DD479FCC35129F4499C19EF98BDCEA9D"
00096     "4941B3756CDE1997C3AFCAE62B6D9E23341E11CD05A7FFF52F5814011A84D737E126410900"
00097     "6BEF5F19E3C6A9C7521B44741A8282755A8F0DC2FA0E1F6CA4FB34D8CD5FAA27E188088687"
00098     "25B9634376137C1BBC46934F83958112D03082DDD6148F353BD1DD24B9F8FD7AD89C40DA0A"
00099     "92A8DBE3608038CD56FFC4ACA35241D76FAC4CAE1211AAD9D73D51C81C59BCE05F71C34573"
00100     "0D3A2C670F8F533A950EF24B00EFE6A3F1354694ABCC6FD9EC4E74DDE1F287AD4F847A297E"
00101     "CCCC39AF029EFCDDDB19932D906B9CEDFCBE0D422CEE305DD05E407340F28EEEA866664D60"
00102     "AF293A45D5D6D5C0000B05F79463DB513ED488DE7BD4EC9EACFEF973B23CE4E9539EFCB797"
00103     "456CF5FD1EC54FDCEE80B39063C48B91A5C2D2BEBC81B9B46D0AD6503BE5AACED2BA5EBE81"
00104     "F630B4E07510356E8229F7FC5EA532B8729CDB819E066A15379AC6942CD4BC5E97C6791E09"
```

```
00105     "8105C323A3A3DA3880D5EE5562ABBA2BDC9906F4486B51ACF8AA4405E9D7A63DB9E3058782"
00106     "DD9AF3995FFB3D34AEF98234A0B3DC62C339325B60706C068F0198BD8FA658396D06931B06"
00107     "9155217690C7F88FD230CDB38E3E48530BD47722FC",  // PRN 01
00108     "9D8CF144C4B667345D44F765622A956CAC4E097AB1CAB05CFBCC6BB68C709503AD9DB09C09"
00109     "C983D46A04A05B6F7EB26DB4D46F868C10E112828B1AEDB3C0074BE0DE3C9B7821BABB4F8B"
00110     "8E24F69869CCD981B09A783BF6A95F39ECFAF25DED6B16F89EA09D3A8413CCEBB545651B36"
00111     "3DD385D12BB72420440C40E804FA27DE029A1E08629BAAB598C035DC58FDD309844F3BEBDE"
00112     "40FCC231F38605DED06572ADD85DC51D3D8B89B4480143D0B75283522354330E5CCF4DE1A6"
00113     "E68047D5B8D45D835A891F2D40C9DB8A76CEB1D18FE2BC38D080A8D97064CC87D692DF2118"
00114     "4ABFDDA7642D0BD6F3209D06B4AE7600F7DDDB71DA751120599117ECCE645FD109CCA2EC7D"
00115     "B98F4177F14DB854FEB314B5D7CDC3385AD203464EADEAFF4AD08DFEF3D21240BFB8EFCAAC"
00116     "1356C72A0F5C61BE03CD2A21A7D756FA9003D562FC4A49A6BE788EC8D80054ACA881DFFF72"
00117     "C2966EECD09F185EDD11218C6696DB14E05FFF3644D11E508F4F1E9C5AB3074FB1C3FB2109"
00118     "2A1C8D5AE05688FA4A9226C3C30D0BC3981933DC8648240F8CB67085F53AC5295428DC8447"
00119     "A1E5A46C2BA86796982C4C6CC647FD8079BC4024BB69E2B226E6F3D0F8A90B4D36DA2AED4C"
00120     "6BB60D318AA7479FDC2031143C67CB4381C27072E12935001524C7BECEDAA9954BCC2AA218"
00121     "E9EC2C95498FD8DF655C015896D9ED42CE7F91CBBA2CC4A7920038EBB5F5CE638F969F8B17"
00122     "9E72AE252BE7E826E5CB53C2E85AAF1E1F1AD8D534F78A681928818AC3154651FFC583DEB0"
00123     "A6A1F40B98771ACC528AAF80D210ADAF83597869968D499ADE9A19BAF341E8CBA20F0E1473"
00124     "BDD898C24C7A5466F9924EC7EE992A2086AF295BEE1F6D0F8843D91180BF2C981C11FD978B"
00125     "23B6BAF7786BD526B458B76A87C31D7C52DFA43F3D362C8EEFFFB3FE5FB3F6E5F34B1FEC7E"
00126     "F1031146F3F609B32677F148F7DEBCF3526BB45582436A3092408193D6312626E46ECFA96F"
00127     "EAD12A234CACE10FAF9DE75EE2D238088146328E10E9ECDBB0B018ECDF2725415CF5A06AAB"
00128     "857403BBF6CBFC350903A982864827988BC805A3484A31FECF7A40D4FE251BC7E487613B9D"
00129     "3A48D3C7DAEFDC49C4B7E625F868DB53A798515A61050978552699EF2A5BF2F13BDD444EAD"
00130     "C9B60B479FDD4633EB4C1062AA78BEF06692DED203819D3160310FD7F2343732156A9CBCB0"
00131     "B50BA9A8F93E339B702670E54BFA6DB2E2E773202C690FB71EB03671AB0B1B02B2F189BD99"
00132     "061ADD23F75F4914067AE638C9A29DD3661C28AE272CE692CBDE6AE880FBCF272E54834237"
00133     "2CBAF6370C7E3AE9648341CE7310BE1C534B5702B0611AF65868F840B6B7613FDAEA21DEFB"
00134     "4F2024487023B02B8B58C9E9F27AA787EE775249EFC40913CBBD69C38538F239B203815F00"
00135     "F7B9CB30DC79E6A0C3E069D109E4A1BAEEE36D354C3D0121F1342F1F4AC504A68D69DEC158"
00136     "D54B04BE8164B48F31BC0827A0379C5237070B6F963741AD9ED4F3865698FB8233D7F49ED4"
00137     "E0EEF3AD927CBAF4FAE183252BC56AE4CDE3E329B1D9C87C6C11429B15B8EE589213CFAC20"
00138     "8A12AA01B4F1F7CC35CD0AEAE217471B3DAC1C279F353DC61994FC45FEDBBE0005D8EC7293"
00139     "85645864EF98A3A417E62F1EACA7E60D4E773BB2E4024D62830F103A7988733DD7BBCF3AB0"
00140     "CD0049006FE2F7EB3821724BEC37EAE44681A9699A025D212724CD98CA3415FE2BD09FADC0"
00141     "2F1501FA38A6083427B662DDCBD0460E12A09072698EC8966C47B8A640AC79C1B7722E78A6"
00142     "C28680F4BB77BBA477BE0A6FAB959B9753217C5708",  // PRN 02
00143     "45D1C8FF162EE106CC87C3EBF6A837930F8CC797EC7A446E8A213ABD239582350636B19B5B"
00144     "E428A9C13F980B7AF5CD7F32630AFE8693CDF0EC0BC2C84F2472F5B86576E8C43136C14717"
00145     "A24705953D392BAC96C1055B782C7941D82FEA357E5FDEFF772FB9F3DF248455CADEAC4CBA"
00146     "2EBA9C91184006D1680E000D59E4BC8FBE2C2F7CC2E78BFA5B60EB292F244E6CF497D5A287"
00147     "432F2520B31B9D9FEC1210923299EDFF043CE077195509E92372F5959AAB4666AE486DEFA4"
00148     "00D81463C388CD05C677BFD4953D2627105B0A776960FEE916C75D53981D30DC689581B7E8"
00149     "E0723D65949662ECFAA6FCC9F0CE8892E367721718F906207663F9AD450AE98D75DF004080"
00150     "FC15DC2CD7A1DCE013A0E547ADDC29A397ECB9E7FA02035327AC40240E2091098708D42456"
00151     "3AB7C5867F3F2D78EE3EF5B658FDDBD49435060CA2EA3D559CDE957B7E48B98DB41CF875F7"
00152     "B3D9EBDF6547B4EDD98DF4B747B0793152FA8CC07C6D9EE5A2002464566D86466C2EDE54A2"
00153     "BF4BBE823049E57364C127A14BFE1B88ECF70EFB81EB831BBF50F6AE124E5F6A775F3F2620"
00154     "E91D489CCF24811C0890EF905E9E2ACD399E13DC81333A54BDD295B872EB74E412E2FB654A"
00155     "9874854FBC3A68C73434C5FC5CED27534B2B13C316205FF4E432FAFC13A7B5B7A7FFA9FEED"
00156     "B5AE69036F8F2955DA124CE5856E8C53F24E609F7D3386DC5212B2E78B5AA23B59D45FE98A"
00157     "A08E9CFAA9D52ED260A36AF07522C047ED43808A39D7019E444EDF84D885A9AC84092A0F6B"
00158     "FED562F3E0D79FB5CF62F98E67EF219FA3F5AEB7D4E344642D3D4B1A7EEA18464F6CE8D4CB"
00159     "3181D9EBF6F4122751B54D0D7F3FC470A91B547148AAB1CA0DF59872120190640555A7561B"
00160     "0F2C11280768F74B1A56674FD5480B0F510491431810D99CECBC6DB85888BACBE2B020FB8B"
00161     "3D78039773229714156494EAEC3A2D0A59E718F72205747D69C05DDF1C678E2E154A1F84EF"
00162     "0CA2E24DC4A6A996F0850A396D2432596EAE84AEC0935B8C25D5C65B52A32722F01D281C4F"
00163     "753EE03EB10020E9FA02462CA303DA39560669637532D381EB78AE5EC0F6DBF6273EC97944"
00164     "2E6243F65FC51F26C6C9554C6C0E3EFF33BC4EAB6A27CAB9383BE7DDDE4218C4998033B479"
00165     "19503E1C9A789711EBEAAD6C0298B3DC563F54D28675260F6D896F1B8D4FD0001C42921039"
00166     "8E9544B3DAA12C31F7EE82EF4D2234E26F873610B76756DDACD24B6132BCFFE735FE75513E"
00167     "D527DD04D7DC6D24059F85706679DCD1474A9DB9571426BE17E6DEBA58B33B708567697F47"
00168     "1CA8B78E8FA73B0E18CB6F88BF9E4F442F0FC21FAB89305484828F18B65F9D373A6A2B380D"
00169     "73F5924F80DA234C1DD87416D025E4E663C96F287B0C83DC92C2164D81830781B715209FD1"
00170     "1A65E64962D805389BAAA91DFBB990D3511E506A8EC101131C5B7284252F861D047DB2C202"
00171     "7DBAAD487ABFE429CA21CBEA7671350618E441F4D62F2D579CAE29D97023A8873869B55329"
00172     "3D9F54D4A929E252AF132325A6E3BCBF7B36D0DAFA1E56A39A5D801FD0D5A41111017BF62A"
00173     "AF8346C7D424FE007C32B437ADE60AAA9540AA5078FE6C3C3CCEA53EE863086646C976FE6C"
00174     "79434A0AA4F53B2E9E2C3B4CF9C9C4015391E27CDFF5C1FCCCC00BBF5B99715A1265F591E2"
00175     "94D530DB14DFD485AD34BBCEA32E5B5D0EED15F88BF5D96D058E6D70BB1A232597E35A625E"
00176     "5E8C2EF5E7031A71F70309019A0591BA0A50E87C839498255A3602C0FAE53166BE5E49E29D"
00177     "24AEC47002B698F80FC49E718B66A8959259ACC540",  // PRN 03
00178     "7A0133D5CC3754D6B259A2CC4EC0298111D098CDFB40549E5C40B36A2846CB4B256672BA18"
00179     "9CD3A05293BB36B167508A7BECA3110BFF339BA06340585DE8EDA03AD244A77F54B7931610"
00180     "B6F9C5C54D688A0526A9B52605BD7D7BB01A63F3D1565CF78ED904BBE4AFA4A290EADFE9DC"
00181     "156E59BAC162A818B6CDF38D2BC715144D44A1578BFC7274237777784D15ACFE80FACAB61F9"
00182     "E58B5D3FFDDB065A00C5D49DF0237EA6C488D7758F1A689DC59DBFF78261016A7C723FE52F"
00183     "FE571F5876FE0ED50FB00A90BB82B27BCAF5A67374284844E06BB1B2D84B1D20228F5C2208"
00184     "CA7E8EEC2E95027B09372A309223A15C132543FF3A89B7AFBA56AA7A8DBE70E0805D3A5419"
00185     "1CC6884D75ED0FC00C06D9D488B0F4816E12D6C2A4324EF742AC8FF885E42100849DA05E3B"
00186     "7C451D43ADBEDDFDD13076CD8D22BDA101F665B5878E321A009B970D1F4C48503CE3536554"
00187     "3B3F36786802E5C53FDD756C595784E4F130904044660784ECD9C9161477F5BCEFC9898754"
00188     "0AD1E86CABD3EC7823D83877605FFC79820F1381DA29282C3C5B3443B6A67973F0622EA5DA"
00189     "A14FA239542EB140082F242958B39014486E5D5632C62C3EC8BE0E09E038C0ABD52B1322E0"
00190     "F7407FA53AE8D761858136CB371AED5E6FC9D32CADC8F870EE833B7120BC0278D9A05554D9"
00191     "0DBEB24ED6F5A8861698D48B4BF7AFE7763C1401572643E246DC65853996B0480D38DB7302"
```

```
00192     "3644093571370CD0B416561B9511BA43CA34341FEF7954C28B2D9FF96EA110E0818309C32A"
00193     "D306DA077CC911299FAD6396C872F3F6AF7871395D7E67879EFED929E4C05AB4C09E8BB396"
00194     "048150A4161D7944CBD99C94DD16CD8E0D8BB73768B17EC02C0D4206AF623037D6F4257DAB"
00195     "4C07B4A6C0B4D2E0C9923FBADE3DFF7FDAB45F4E6BC5A895FAE4F5BB9EA247F2D4446E260F"
00196     "7988C452203EEA1DFA64DEC2DCC090BB3ABE13F6A8718F8DA2BE551407B59B8EF1806A6552"
00197     "6B6B872CB8922BB929F09341554A71E69B41B60987FE3A5E7E3424D947455083A827FFE27F"
00198     "B5BC5365C80998DE01CCDB66213575FB61B3E6F877D0E2E4EFDE4467D9F07B6A28148FE2FD"
00199     "6EDC9202F55FC855D0DF8C49E244C40CC3D95FC06C6778D397461BD157F4A0FFD915799820"
00200     "D55F52C96AEC0CC5A3E7A2151A845EECEE78B82ED9A217E326CF6C49F7D31C4D8ACAFD827E"
00201     "6ABB760150203448C000819E7E0B6E424C43A5500164CF128D686B4810D98384806044A8917"
00202     "92987FDC549D87F95BCA120AB84FCCD8C9F93F988C87E79599F3C1952BC0F7773BDFFC50B1"
00203     "9BDF8E3D8F52D887E45B643297650044E80124BEA0ED60FEC4449BB3BBE394CDF7CC7AA39B"
00204     "C1A5023044F6A843186C01EE1BF5834EB5401AF7905FC04447AE00DAC50B051B432F831FE5"
00205     "AAA7506160CFD7D4639C489ECA447F4F993AF0503CE5EF68A837FCF85B85993ECC55A9A367"
00206     "3F8F2C5CB8D3DD4C60E8421E3417EA958EF87E0764B061A39C32ABD5E0E3A712B54B0A3E2D"
00207     "351A0E00F4E901521C63C1F4ED829F6E259A1F720FDE96EA9CBB8F7BD7485531A81A49CDEF"
00208     "CE725493A04B5EAAF7411DEACDE5A95AB6C2AA7BE3269F6AE4D166D8A5FD5264B135FF8361"
00209     "FF75B2FF22A61905A349C6B2C1DB2BD7B385B8FBDCB2768C7926F138D5F8107111563CE527"
00210     "322AC42E6BD42485668106B8CE91F157E0C94448869F7AABA255821DF981CD5298D40378FB"
00211     "0E33A3DF8A037BC21F0AE268E69F7CF61E7E117BB463EDE3C7D2EF95987C66AC3E5C7D79C4"
00212     "4A7590BFCF998683701DAD7B98731DBCB455E61428",  // PRN 04
00213     "64D4236F326627BC08E9B2B96C1A9E5BA2631DCCA3F7A5B63736E4EA8074056DCD6BF0E5D6"
00214     "DB8845D9271C0D706F972AF22E652E3B2A7CC482B125EF8BE005F25D5109F6F9DD84DF966E"
00215     "2E0B8950FE1C01E2DF15205EC48BECEFB32511DCC39678F9DB08ED0EFF64C7B5DAED1DFD20"
00216     "2F63B6EDBCFB7E4FBB431718BBFA2B65594D78D3983B0457DDDB350AF2C1B9743AB9EFC260"
00217     "A78C3144622C50D528B7D47DDE46FB9DA33D1E7DE6D5829258C2F02D54085AD0121A1BC833"
00218     "9A2847F6F161CB6EEFC5E3FFB12E6C77E5FE10E0D4B02E58142DBE5BF900B7B64EA79D520A"
00219     "35506AF4987E67BCC80D00FF467A0889A14B85B4745888B09DF7B4D3316D9B4A9B17C4CEDF"
00220     "CB0A55E2AAC53B538705673B2A2EE4ABA914E20ECAED730E3D5E272A65A74C366177A71109"
00221     "5624A679C9300FEC18E20E65DA3645F23AABEF7EB7C7FF49B289FD2F44661F6F5F689B6E06"
00222     "EFCC99F8A5EAF558E10F2236E002F7D97C316698D6833792C456F812A339166E9FB3B3F617"
00223     "34B98AD2DC9A484A14A6025114D072B68C47DDE97322E213F5D6A93762A45C4C73C7832BB3"
00224     "1209C4B1F0094BD24BA23CA9378FB893AB078FA3BC4763704EB6A0C8EC965690CA5B858DAF"
00225     "E623AB88ED39154F46B9030462DEC9070DA9DC34063C1CAB3FB84D60337F1D6D95D1A17365"
00226     "0C96128D3C036D0B72B4D2A298186CCA8E1F386E1F70C0716F0BE370A6B325CDFDEAD3CC67"
00227     "E021E9D4C839230708DCCB062096F32C4DBE3C4876A7D1A26072673A6CABDD8D65A5E91CC5"
00228     "E973F00FA67619F749930F9D40B767E434D0955D47FC17BB37E4B7CCE63B1D666AF5D67FCE"
00229     "5FE5469D3DF6B6855F9C308DDFDF6733FA7B8511B25193C27925F3619F5F836014A64A2C8B"
00230     "783A50C6B3001001D621CEC582FA4E52C2B916A418F49ECFDB5ABCA40E1A8D38FE400FC2D0"
00231     "C185009E85546A92E829DC1AD2A7AC3DD23EB77F3D80015BDA4135B1194B10ECFC87C1102B"
00232     "3D366C5460D20D83E778142F1A4EDFDECF3D0BC448CE24A49C609B035A99B64FABF071C58D"
00233     "F592BBC625359BC23A565F4BB077DCF879E14ACE87F709D276982A3D10B23A9538A6A3DEEB"
00234     "BF0B712DB4C16B0FF508F3E4AF4AA759738302D24C6D13037DC81B0394FE785FD14A322BC0"
00235     "0ED95E0E9D527FE5316748DD893DB03D5C4149D471DE98C386A2E9DCFF05181E837A1814D5"
00236     "4C9CA708CB5E7A10B0FF4F540338BE403E4B9713015A99260795DF5832CCE8D5F763C68BDA"
00237     "51133AF1D012C283E8F4B1DDBB4EE8C351C4AF86FB436571F53B2E80AC04BE26EA617EFA9C"
00238     "5578FBBD81DD1809EC5BCA5DF4A85B24970EBD88D9856B2136616F85A338771BEE80F20F6E"
00239     "5652FAD00F49F3C349061DE3052841992EAA04CE15DB6C48888BD3CC5EF4D6CEBF00431CCB"
00240     "7C5289D79E67FC6CF2DA88AB60CB65D0F687455A535C33547553966D197BAD9778542F6CC7"
00241     "182289AED366ED1CDB0C5EC1E9C0F0E21D4D2B8D01EF4BA394E0C6B05B8EF42D81F773D3D2"
00242     "51D6264CA23DA5CA2D4884544FDB81A243974FEECB416D0748449BA0CDC69E56D666FCD24D"
00243     "9BD7C322CBC038CEC658B5253227D87C527473F34BBA3E2FC18784F864233FF8528A874A85"
00244     "03840AAD03B63F79FEAA6C07E4467C4A04366ACD35E32A142DD76A23417EDBB8004944A1E5"
00245     "1880CAA97937493EA5BD5A41298863A71D33156CDB942C166748DF66B80ECAD27675A673C8"
00246     "692805B42E97F1B28DCC9BCB58FDF8BE55E2034E73E444A225DDDCA5B46BEBA51AE98A2799"
00247     "0F622B32B2C337046E459B173D94BBE5FDAE31FFD4",  // PRN 05
00248     "23300D5D80ECA6A471DE5CBA1D29B7010240D95FE341A62FF8175ECAA9566011AEDE6EDA0E"
00249     "F2CFFA7BFF9F4C9AB2C97F6554BB182B23F2772090FE7C4ACB8BE7427E7A8535DA3A670341"
00250     "D88CFE694D1CF40156E9B0557EA317A0B3E21F3A629D1CA971482B1A54697BBC2020BBACBB"
00251     "B9D1E67DC33F52C4B446A36D4331FB02F19BF1A647D05714450E591C65853176A04DD561E0"
00252     "933CF3F24A2E4707C44FE29C5D7706723FF53ADDDC67A67C23769CA32876F0CF31D233D352"
00253     "FF6A7277E5A3A7578A6F2A76456C0AA876680CDF2702B114E02D22D9F59077B9DB2ABFED67"
00254     "3158EFAE4C23A75FC8BE701D973169AA7015297BDCAD4870D4F152DC556A06BBBBAEDFFCE4"
00255     "0E6BA4F0FEB154F32D8E1492F74EE7085937600EE176ABFD6B8638E983EAD26C63805B9874"
00256     "5BA290813CB65CCE33DF6B98240E571DEC4BFB2430C4B8FEC23C1BD5C3E87E0E746BA8A772"
00257     "2A6A660C5095FEA8E1C4978B966F487376DF42C9668E5D3102F123FDCE7B8D80E8BC84AF0D"
00258     "91E9355FF6F7482BA74E0BBA7DACE85BC053F31074151566334B4DDA37C6C51BB947811F28"
00259     "4B671FE53FC464CC30DDE59A3D9CAD26822183E96D2F4BE60905B94BB8B167B734A0F0B26A"
00260     "D0E5083A34E68A620151966D073293E5D430D0726C0FE35BA3F1F44D851D374ABD80E826DD"
00261     "84A665165B3B82D540CDED9EF60CE48E87802F16F5DD89163AEC8E9C523E9F99AACCC6F001"
00262     "26C4C8663A7D64D919EC41ACA337379496D0CC876199A86404CE5844F8E2CACFB00A985B92"
00263     "B7A393CE464DB19E0B5FD65BA92D8EC1164CB1ECD994F82396F6A52BE2B66C8780A5AEFC02"
00264     "93EE437E7FBF9BF8880B61C8313C3865561A4287BFCABABB90B3A11F6AA57C2B2A6047F131"
00265     "6AFE1DFE540C4F1D5270F3EE3E023E202AA530211DC51F8C2E87636C14EC8300271A6454A9"
00266     "24A8093716756F79780D94DC6863C24C80A405D6F52DEE81810EB850C1F60732AB24209773"
00267     "F66B2D2AEDEDAD6FFF60902910D4858B0F706414F5779590F2DA7C249E5DC1484EC40EDBB0"
00268     "1920A5175CB9D74EFF957A61FE3E08DD7C5DED5B299C03299F25E39B161E1DC1C586E39D0B"
00269     "AF38C09C2EC0B22589AC489C3199EA4E66611C45A68A7254D292C78C3978C381F297D7DCD8"
00270     "C0F7646E7AB6DCF155B67F10C3F915F9B3AD96C21379993D8C5D6957847EA81B3BB4BB0863"
00271     "F0CED12F5CC48A4325CDA65268110C1C156F845951AF3C3C90280F8883CE0236FF02DB0CB0"
00272     "7721261432C7E0D479F859D8EB7C433F67721B06E001498656578F0E3CEF2A6B941519885B"
00273     "BBB03F33DEEA07802226AEBA473CFEA6EB894F45D1BB937ADF5180F5FE22857CE0EB75D251"
00274     "B02D89E5502560A0B6B012C191DE9D62FD28CF503375F2A1FA9EFE0E42DA81A6EB3A6DBA29"
00275     "9726EACC6F3BE91ED51A25EAD5F3E7067720F7D4BB72F8BE2DF978C46E1DFB4B0EA17BFCEF"
00276     "EBBFE40C66ED1F288BE08D6CD0B097C7ED1205E43F8FFB7086120FD153C47272188799D0F4"
00277     "554E9A4131C6B1460077A99E8198B3717AFE5E7C95D3F49B3779EFF9E935FA63A6F881F039"
00278     "436EBDCA2EBE6FE00109B658BA5555BAAA4A401D1FFCBCE0369798BB3BF8B54FCEAA5FE25F"
```

```
00279     "31AA02208B0F070270F9E043BAEEDAD4432B1C2DAC9F7B4CDCB52965EC43C2E99764AD2613"
00280     "BCBE468C9477E64B8BDCB64CFCE01C06EB66A15FB034D1AAA507AFB6842AF66AC8C18807E9"
00281     "8884C6A780805720718A3A4D1A7B094846C55B0808736199CF4EC3F66B713259CC715B22B9"
00282     "2AFB1599B7AD539B188E99B39F9A92987D0ED15C94",  // PRN 06
00283     "91CEF241899B4DD47BF31F3BBEF20F5BF13D9A98D3F133AE61F3E4A87A299A6B115B96DA6B"
00284     "2811414D204A49EDA0E1A763E1EAFB78CE05C181DD0947CD50276A10D62543A0ECBA57DAB5"
00285     "DC794AD7006A520B419533CE8519F4EE4194B58B2E36F9E9705266B6E304D1DBC6B7349104"
00286     "5548637449E5C657B263CBD9577AD8DB7D5A5BB5DF43A869FF91BB8706D4E81A4F8243C214"
00287     "D9884104675F2BCB426CD785A28F4856E363E1ECC974325487DC40E7304D3D7CADA5DCCA6E"
00288     "60F4457DA181A39801D35F20E1DFA05BC09E7699FA289A87EAB311BA8700AEC0F90950E5FF"
00289     "740DAE7EAE7FCAE8B0D9FB82ABD25BF46FAA635F570C22A76C52F9733D5BE64BA67AEB5D82"
00290     "6288D03D8A23C5BD3FD6DBA68082B7815044E24859865ED557C7DE8F866823CD4716E573BF"
00291     "F30067F0931D0AFED0146D55B0B2A3D6C57481D94F61279DC4D2C6DCA1CEE27DF24426E55A"
00292     "B4FFB87068AC8AA6F286151356CEB29D993B4FB26192319C373A1F0F58FF5949783E71671F"
00293     "B58A683DC1CFE166FE1AEFEC8A41C5FC06A31F4EFFFA3EB7C388E1BF99667A5D2697F71324"
00294     "09641AA812EE84EA8735BB46E5069A7BF1B2618B56ACB97C368AA7BDE1BE0F0BF286B6D08B"
00295     "42702F1EDEF408BC529B4158824F673B30E3D3247FA59DCB83D004459CC5EF87BD6C7D188F"
00296     "D91A2A7FFDB008244F8618ED46A73323E9C7F9D5A83298F8C81702FB4523AFEA589D4CC269"
00297     "BD226D04689F132F766770FF23C832EB5695713AB99B74035540031AD1D9D9047D72B1910B"
00298     "C320A65A9F63A40C02FAFE217BA0C51FFE24E96DACE57C1318351038FBB81E76EA34853484"
00299     "E7ECEB672A4180A3D24B6C692747E9D306DD0AF275EC0581A30AC2E80EC73B04B1EF83160F"
00300     "D88DC43406CD715EAB1677F1422FCAB85D1D1EAC64F698B9ED69FC9F2B5519AF9153D5AB72"
00301     "2339434C780886B81DC2878B27177B4616450BC1DE99C49C965DC9257BC78ACA052E998493"
00302     "A0CBF7B6AC86EDBC2627E43E7521A9DE7824DFA7DE19AEEA5892685D50EB10128B3D14F3C6"
00303     "70F639AF6F2B2F6168A1E551A65781A418ED697E6AB3CF91EA2470995A401B08380CA58B94"
00304     "39DDA7BDB7A92E91016567D57D57D5D0DC4D8E3C1490CA1BEC03FDE065C7B7F34DBC40BF70"
00305     "4149A0573C55A0EE4C415533829EEE71BD8C392FD622572DE009044E937AEEDF672886CF4A"
00306     "36ECFE7B0F8BA9CD9EBDEAEC40BD74AD34CCD2C7DD07F7AA64EBF4DA9C8810C05F55619B80"
00307     "8127CCA00D171BFEA3E1EC0895A22218DF1A0514FE5D16DC416D9FFEEA8EA3BC9FC650F678"
00308     "EB731E70A6110E138E059371D5A10BEF6420A0C8BD45F836B7D310E732BDD0FE57D93A6934"
00309     "317FD8E2449BA0F0C6E1B36821B62D1F80B9CB6A5B407B6755977C06BB2C3CEED0B14D4D73"
00310     "A3554FAA54EF2C350C388B230B30148605035E4F481A769564E2814A160A67E6F0DFFA7814"
00311     "D0C12772C0496737C52016A93C6753C82FA896BDFF3BCC72105D6EA5B52B3810004DDEDC26"
00312     "6F1C90CACF49BA7DB5016A4E485F06355441CA204F7D589F2A3159541F991682E6AEAC0C73"
00313     "59B3282A04B2DC69BB0AD6CF49C48343C8A769D3EFBE8D0814E72934F7CDE698061EC68FAA"
00314     "B39016B96BDC5363D2B53363A548E6476647C10F556114698285 64C189430D351CA01D2F10"
00315     "29CA3CE1D2853E8E595D46C9C3EE9F7D3EB9C72A25F7B2138063A0F9186FF41CA9C579DA61"
00316     "772FD60408668A877CCF6B65FC2B88935822A6DDABB97D19F7455A9725A8DF2B8E6B28D83A"
00317     "CDACC66F02370575831490B8D8838D4E56AF36ADF8",  // PRN 07
00318     "AA82DC5072A45A21B7880DEA3E2691FCC22EFF3AC815A2576A7F480ADA6F8426EADB4A96A1"
00319     "7EB949BC049A646D46926F0D69E0B3E1911D2CE652FF4D9CBFCFDC30065FF4F779DF896D38"
00320     "587B297BE8E224EFB1DFD04BC2D22832B2955A250B7D0448AD9C0A76DE7C33A4E2B5DA1B3A"
00321     "868852B7F04D848EA6495BE8D2501BB47F24E8DC4B254DC56BB5A4760DF62BDB229F3DF0E0"
00322     "C06274A63A28D11F928AE2DEA72A9974E3C55CE261943F78B771CAE7B5B98642A01DBC0611"
00323     "1B137774898F0F113A4DB23BC240BFF147568493FCF0C3D767D3548A1C0EF7AA4B6B2DF566"
00324     "C3E94BF9A183264F63E4FBA58B517FC46C4F0017F8D763A54D044C644D6326677A425F0F5A"
00325     "2A7C8796B9C9F8CD0FC49A1F035BB6662C0CCC0E9508AF8D2B834B929D80096F9BFC922BBF"
00326     "1BFC6F101317EC08ECC1DB52404B30B153F43C266D3E64FC496CFBBD350668B2AB3B596B5E"
00327     "0F0A00757E0D4A771F5A386AE86F5105C1718F7E93CF00F588C722509A03597B95512CFC70"
00328     "5F08D3B4D5A3E23C56EB461A719A955AFD3EDC8329BC97E3EC6BAB24D97E16A724ED2D4C02"
00329     "B2371FA3ACAF49175A441C3FDE53C2DDA440C18BDB8A6F27805F944EAA7991D404CA74544E"
00330     "2F3A669DCC4052B6770AAD9EE29A6EF7791C6DF93054D07A2B3CB2327B0EBE7D9836246399"
00331     "61AC4C3CEC2A84DBB3A10A830DCAABA9600DED87FD5CBAF4D5F1DB357ED42374175257474B"
00332     "AC200AC1B6A5B87D2FB2092458F51185C380B8BFF682C5BA1D2CBF8BC02E5D485FD811797A"
00333     "ED167DE6ACA66B927363D12EA405D75A2E9182B1FD30ABA700C5683611A24EC5DC453F523E"
00334     "AEA44C6E06261A98346328ED9E86CA8F7EA79EEC551F36836272127A45984EE165BEF8AED8"
00335     "0D26DC45E34E1794F56C1265DD92B4078FD2AF2F13981C08FE27C55C3C9238BBDE193A956E"
00336     "3F834445D949CAD84D3FF0FCD511C6C598735D5B3B07BDD7D437ADED3ABD6EC3171F373588"
00337     "3C9F511A06F4C6C7D0B137DD0D57869B8FB1375FBFCF9C3D08CDCB12B8D01614AA3C965550"
00338     "E5862AF49B04410D25BA4BE86FA6B0B9D9142461AD3CB4BD6D902EFA49632A78F1463619E1"
00339     "309CC89452C2B453BF9F08A714D67ED90972C62C0468CE17006F9B60138D28DA6362670BA3"
00340     "048CD8D099AF193619AA8384A1758FDF4A04CE56ABE464A66E913DB7BBBFF7EB2ADBD47E63"
00341     "47D7054CCA13000DE13F550C6263AA4135A2E16F1A2A58BEF962113C209CF58CE514FB51EC"
00342     "162D05A4DF832697E544037CA18E62A267D81D539F879F50C654E74BA21B47FFA5C704FAD2"
00343     "147EB3DD8617DE98B3ED4859B605310A777F3DD161F4038486F0872AF55FA3610EE0D68C1D"
00344     "51E0F91EF7D9AF4ED01BFF53FDA16BA3197A518BAC0F82F8895A2BDD9FF4C3035379E870C4"
00345     "9DA1AF18B3668792B6640E687BB71F13DF1650C4E1A2CC487C247D1D1356EE16DB8F973634"
00346     "65614E9E63F36C853FB63C12963A8A5D98B52BE8DB31F0954B35C3C749A62C2B34A690803F"
00347     "A660870155062251 85C5D6FC101D1494C3A7961ACFB4D9B905323DAADABA1DCD2DDC1F9CBF"
00348     "ED7D726E602578CFDC1519925F8DBF9AC5FB4E4CA723BB264B5D106B6206574C46C1A49309"
00349     "C22E1935904942C36148F764B59F10A5CBA0C9397F5798E200A606EFC75DEEA1FFB10A8539"
00350     "8E5BFBBC6AA3619A0F611E591245D03B6ADC0C50440A1B1C236ADBA933B2BF84D2C60C807E"
00351     "9F52436904BC62B813FFF8B9C824819C760D0BD636B10572C781B112CC0C603A260AB87986"
00352     "B0280FB913A9E0BCCDEE347F0C744BF8299A9099B8",  // PRN 08
00353     "F2A17D19ECA96F67ADACC0D83FDCF235F07832E8E78A44A53F9C4CD09C3F88DB2267019631"
00354     "47EE0E1AD0F549E11BBA70053A105BD62C40DA810BA37BA72D999CC4FF9BA4D01FCFC670E0"
00355     "D7819871D06C4935C10B85594CB7202F037B25F85141A580AE1CB3E0CA91AD73C6947D2492"
00356     "B8FD2F889B1421E04FF0FB7866218A491D9EF6A35DCACC12CE098EC575C697F5F8D1BF9178"
00357     "B3B36C999EADFFB5CB3A41DB672165CF73A7874FF68B3B901C4E9B8112EF4FF9B9308742ED"
00358     "678DF30E2F7C40BF922099C1A95BE0C231E9F8332EBEEFD99C6DDEF25CA5EBB2985A968991"
00359     "2616815AEA022BFBB87353B8B799D64280A6922CB09044ECAFA51037AC2FCB11516E9E286F"
00360     "64B4CE3E30A55191F24AB53F2E10E5A8920FFBAED77FD4C476E9AEC751763D68409ED9BF7F"
00361     "435E53D401CAD787A50033BB8547C910D90DE100FDF6B904A1F529830E3A51919299FD4476"
00362     "F02C3600AD5D42E3E154AEDCD1C99C1C5B531FE179860EC1EAC69EA0381CD4DCDE8716F94D"
00363     "6510F10FD1915864BA968440CCA6D0D5AD8983C421079B033B56FC34481B9F27188F829F91"
00364     "DD73F27BE8E0456B0F75DBEF40A3C67F274F6A50CDBFFF9798F8DBF1A6CE7158B577550E40"
00365     "639D38861A3CC4E060C9E5BC1F0D3760FA9A89C2CF0A23505F4A642F5EB0A055B96843198C"
```

```
00366     "E20133DFE022AFEE043191007276427DC82E877BDE27E20D65DFF8232E9E4DD786C23D4E8B"
00367     "5E84637A22AE5562651BA45369947C159B641710C98494ACAD48B4AD6AD9828897437AFACE"
00368     "E442DF2330669D5F2C6A9893E08507ADBF104C62F6B4D568C5381B28D5162EF0FBECF396E7"
00369     "C622B54A7864F9B31CCB396A0DF82AB86D950B4657237FB769122BE6B783ECE3F798AF68E3"
00370     "54C521C77735EDB97D580CC80877ED702231CE2F8B73262AE39EF94E84736949292E065515"
00371     "D40A16BAB13EB9437D30AEC44AD8C67E3AFCE9AB377753BDC481E8E79EB10F89A7C4F7AA24"
00372     "E6FA48A36DD1CFC02CF3FF6DFD0CC1C4030C312532A70B60F678FF2B3586D77DDE74167CFF"
00373     "A62C12F7D0086A6FA59B36C6A025DEA1EA0BFA9861BEFEBEC6B2601DF225E29F36AC97EBDD"
00374     "B21D7E8206E1EC42F3B9CB314E7AE6464D54ABD53D090DE83B466FAF5384E0FAAA6E67F2FB"
00375     "D6A72E00D9274C9B5C768DB4AA4281F25EB2DC6BBE0EA85CE1CD31B32BD1971D6AA20B6C68"
00376     "D66426F09C0C1128A679AF06CAD490DCE3A2DB50181278BC40DF9E094D1D09B281CD1EF1C3"
00377     "69E5927407E04918BD21315E12583D7D845933AEF186DAF8B6609C3CA9452857DD2112E862"
00378     "27422491DC7821D7E41DF759BABE6ADD11A4C771CF16469F86262D169F221D97DAC9686B6D"
00379     "A29568B4A2D2579C76CBF948C321F5158EB3FF5F6D4D6D68FA255C81F62A8DD605507D0C6C"
00380     "9D82E6AD258B32DE4AB6FF8FCC4A8C237E270B9673C208E56AA6F4796867762399FC8549DB"
00381     "D232B2CD0F9957A8A24FCEC6E8B3DCCBC67085C542A134F3E80A3CCE8D3DBEF0528432F2BA"
00382     "B2F5DC89D6015BE7F24B2B1F378EE75222EF125A5645AE37AFE6ABDF84CCC3026507BAD9CD"
00383     "8C2EB4020E7763E0F85164734C851532FFE705C111CCB7D738BA8C0D29C5A0A50250D0BE53"
00384     "353A6C407D4F7EDD62F271F099528CCB520E5B6B5695D3FBE628FF65D67CC0D1B766149B9C"
00385     "89D3C770AA133FAE2BF3C9846B30C46584E9F93A8216627CFAD84C37AAD7986B8D69F83392"
00386     "FF8F98C828F8B136633A09718D375327B98A0CF8B7736D229499A52D91D19CBB896A5BF413"
00387     "1691682741797A25FFFE6AC60E30C79385DC778C90",  // PRN 09
00388     "3D84AE0F7756BCB2518300DB110A07EC46869AC35F1B02650A36C3CA0350356E2808B84A0D"
00389     "9A8A648F920B7D48AFBBF6CA9A37F673DE2B6FB15B189FBBF2444BDE8808118595AA995AE0"
00390     "DF53F5146B011A0AF5B7FCB4B3D3EC074F2DF15CC9708EC50EBEEAFE06F7A0235A50E2DF7F"
00391     "6C023B53819AEA87318C4E67C4742FDEACF57DC539FBD5BA01F64CB45322212C707923303D"
00392     "96E58EB95F7392292885391283DD94772186976984AD6D83AA96B06D68F58FC2B8FD9D61E75"
00393     "337C6815BBFA8B76833FE1856B714ACCB6FB1B5E957DD385658AC760AC5811E3BC079DDF2C"
00394     "E30BDC21AB8D2C7B56D2F6B384FECDE4EB0206A6C00282950C1B72B3E464570364A919812E"
00395     "86BBFD0E1CB2DC98D6FF42617355F2131F72A11F50618CF21A4C4961083FC6BFAB63F54467"
00396     "7428BB9F10FE341B947C41AAC1B367E338CC2AA08D6CAD1067F02654794571FA4AA11E211F"
00397     "5E75D194197BE4AF6543009C40EA86F74AEADAAB134CA9E35D79F7CD062B53BE2AD86FA7C2"
00398     "E30D9222A494A25044153420AEB6C8FFC4E0D4C514B8D9F5DD922D8EA387003409539D0B40"
00399     "210B05C9209B008D3E47E8F84C376F46001FFE463F21A92E34D3345751E8E94205EA06FAC1"
00400     "6292B8C6D4FA028867AD4C4E8DE0D51BB7AED59547707CFBC2CF3A9235C29EE3A3873B96FB"
00401     "75DD50E758CB2C7196BDC09336A97756DAB994C46D5BBB87F9D80008D09C603A68022BADB1"
00402     "D2F6B98B6B76CB3C0EDAEA07A23065B93DAD67BC59C9BDEF7BDCDEF39D9AC04C9CE8F6DB4A"
00403     "925F77BBF8150AD478E3C6D7D32512ED75BFBCB82C66D505CD8BC5A5679014B355BDC66F36"
00404     "5BE9AAF15B76C0CFC0B51E53F86BB4763F0302B448E84B9792F34442743E2D979C5133514D"
00405     "4BB1DE5E8AF938C4CF158336D04B6723F9422161805D5EB059E55C86D0C461399073A705FD"
00406     "6E919D1760B2D6860D65665B486DC0AAD8F7B3ADA9BD681D7FAE4185350BAA543E9F8671F1"
00407     "B4BAEB6D18674516033B2E2E4E427521C290A6E44E1EAF9974BBB360FEA3535CF46D68193C"
00408     "0E208AE5B2F85C2049C906725C92F598E6C0B67DC5F8B7C48CD411E107F4860CFB3C4229A4"
00409     "4694ADE0E31A4FD70AE5702C0A7710678763B035C5C6B4B9214D507313F5ED85C444B98278"
00410     "B40446C861057AB9B897638DE4739088FC28A577CBD52A3910885276A2DC8DA99EC74D30DF"
00411     "707BFC60663C869DA4C47BBF4D64A789425C5C894EC2B512C3296D0070DE4942113FFC0B13"
00412     "D03515F2B4EC07A0E3BB7AAF08DE24779F9EEBE26CE7C316EAF47396E538672BB646E1B201"
00413     "6356A407B8828DC25D8F8A8296B4F8552901317 34C94CB92CBA5689574CE95B60988C42558"
00414     "DAD978F36D67BDDDAA92FEF21282683EBF6D59763384781EB4D3431DF5BB909ADACB397D36"
00415     "D55D2ADEE537D7B1B942F3A815917A388B12285689B40C3B45C04A591672A2B957F924F1CF"
00416     "6049B96A38E304B8641DF6A079614C9F9984FD9C9F0A82ED6E440164AC484252B45147929A"
00417     "58B069E663663CEDF0FA44FD67580BFD23BF77C61B95B533107A25684FDBAA4C1285AB568C"
00418     "9FAB387C6FDDFD5EBDF5AF1CE094C59630E19F621CDED14E3847BDE9B0EEC64C159F23C779"
00419     "C4B801635549003 93CD7F1B885F92F183839CDAD7801DC7FCFD08B7E08D23BE21BC80F8655"
00420     "A2F3BD0C483B2C1D23D447682CCD9CE098AECFD7BA7C48F5BDEAC8397220A7852077A8D793"
00421     "3FD353C25E37DAA473E996A5920DEDB80C26DE4FC3AFBD6ACEA16DD3208C9384021D9B1FC1"
00422     "C884319B59276D48D8BF5A0A37D84B8C579EE0D9BC",  // PRN 10
00423     "446D385D09A3464D1ABDC6D6CCB015280BBE7DCF00078634371D7F327322DD720E8DBD7B8C"
00424     "D97FB9505A118B469762ACA10C4E135ED782FC905EF30F9FBA82F11AE679431F270351F70F"
00425     "0508BA998AD9A71D2BE05F55348B275A0E537C63699C96DD8AB8E0645CA2606E35805F66CE"
00426     "586C81B4A65E2EF036697A18DA49C36CB2240449757710792 4DC02FD87072490D7EC8D5B37"
00427     "6D0C3370F9753E0E7C781952C03D844490370D6853FCA49E12139805C24FE9C9914E3CF914"
00428     "64D332C934570DCD5D1C3C609C6984E28C9BFCBA3669B75452478DA4099CFE0F6311FF782D"
00429     "1D0123A120F657CB536A3810984871B69D25913C305117A7E51D199A56387C2777E349748B"
00430     "9C2C40DECC03DE7235F7237CF43FD421039743CA72361DFEF2491953A3C3F0091FE3EE064B"
00431     "838BD73E9E6EDF8FD90235CDB374892BFC42759792E5636D031795C0DCC70D7DB86C161FE7"
00432     "059CE17B6926FE5D024D6B1B9DD5608839DF85C7802F5655525B633BDAD0803CE14649095C"
00433     "0A7F870F66888C072C410E0E0854238BA7FB259901CAFE3570D398B621518193BBDD75258B"
00434     "CB53AF2789EC66DBE2311DACDC1F9CDB6DBD4BCDCD15F9C2E77E0D2736CD191D2B83389A71"
00435     "B1804A0FB5F2480E38343CA48EC6D1830BA3466E53FC51C4EF05F4D10711A78AB215650462"
00436     "E9A26D9301EFDF15A3CD5EEE14DDAFEFA45171BA3A757CD6FAA5887A5D5A98045BEC7FEB33"
00437     "1D01A997282910B6B946CADF0AEA57973520EF0561B233BCF798805B3B048751EA086FA71B"
00438     "CAAE15F8BEB837CD2974EB3E54D3B6D3F99A51F88313FB64218610CD1831E2E6E18EADCFCB"
00439     "E72BE5818B4AE4F3780BAA7D8C4A55DAFC10ACA3EE7AB7D1337506C485C83EB0C41748C668"
00440     "705B4D41D926221D33C31BE7D56B42BAC7E9240FE4C2287B5B41FE88F9B9E19970D0E4569D"
00441     "BFF167EC42BAA2566676EB055B69D159358D23DABFFC86A917490B4942A220038CDADBE24C"
00442     "335DEB895F3472F0D22778BC79B169D1EDA9FD18B31DA66F0246359DB3EBA36269EC5F92E5"
00443     "A1A89560C7DEEFA121A7035EC2501C553854724C5176AE60F87004F6FAEA907B91D1C58837"
00444     "C9FE25BD9C32F66257BD78169B1A4B8ADEBA72B5699D4C2D14F67BB347EC8A96B04278AD9F"
00445     "97A9B40DBB56E7593CF4B3C905108F10968F0AE6A2E7F1D214FC7B45637A256ADA5EEB70E0"
00446     "3C02DA2ABF86C36E581A31032DFFEF4343C3E59EC7A2A715E9C199DAFD3BF8C9B43C1D016D"
00447     "CD33E2590A20EC8B657C08E5C0C0BB4F159A07206504780715A252D1F4F5DC65ACCC43E3CD"
00448     "C255123A77792685C9B68AF281863AC89ADBD5F778733E95050ACF781FFA386B3EBCD633D6"
00449     "E8E6D4A42EA85F005AA80A4597D40738A8E8823FE3C84B10CEF044E9651E45D1ECDF59C537"
00450     "AACA58D2BE42655259DFE6CFFFBDAED252254B732EDC93B4D610C50DC81B576C5A5C5BBCAC"
00451     "0BAFD7BDD7771EF1A19C29D7A82E448AFA07E7FAA8651C11D1BD550DDC69F4C9C6A4443503"
00452     "4D80B47B512155E5141378A94CD277D4BE11B2A383B1539AC9731079624E81D3A3404F5325"
```

```
00453     "F5D2EFBC0929F510529C84B1BC3E7C08F661640E55AC49452C00256BEB49B7A52DA23EFFD8"
00454     "5B8C303C6EEDD2202768ECABBFBEE87FBF1DAA992C8439602B1645C462D3297DDA1AD94161"
00455     "7B508E481A28D1540212CB281B5388A3691A26AD7D3CEF52FEEE9F59F477EC98AFCFE38034"
00456     "B762B9D4B5DF3BB312758C4238EFFF94BD16D9C39655CF07263CD6DBA96D1C0EA425610C1F"
00457     "C81061FAAC5C0F8E6AA9591EDE2FAA1AAB8F76EE18",  // PRN 11
00458     "C514F282EE4B5E02A90A2AE126DA52A9FF5274735ECAFA74CB36D607EF299E0227E94B6CC2"
00459     "B78B1F71D6B5FB8BB4C78C09B3A6E3ED6CD590AD526F74CBF869085F2BD3A9DF66D7DA9DD0"
00460     "6FC6748D8FBB78D98A4D860B362BEC6295F34A6A16A58BB21C9632F8B3A64370B7A7547E32"
00461     "9B39BCC82D17B82F6E4ADB514444F62AA800C97B748D73B0168ACF2FA898C262820F898171"
00462     "3D00CEB8453E3E9E452F5EF41452A9C8CD1F1260000C4CA9AF4E6E629B13DFF203187C0DED"
00463     "A08315AF24181015D2E68D6AA4E7E938FC4FA9A8B80CBF49E3A9F27953A05FC7609146EE39"
00464     "F09FF485015168636D1F452FF311D4D8749C31AC8420865A87A707E540D7FDF0D25F63C7F3"
00465     "AC906E048281BFEA46263B4350BAFA8B5A27E0CFB23DE145622BC2FE34DCF4E1B4DCBF60EC"
00466     "D5426C27B3559B73A59CB1B71587C685542F5E6DF9FE3C5E75A8E9600E89217343D29C67A7"
00467     "AE80F9A8589D82AAE7B2A1B8D6546D3CE2482DDABCE7FFD3BEB2F1009297A77503E7155ABD"
00468     "4692F1244A49AB32628947A681388AD75A750AB92C81ADFBCED0AF166326FADAE6B7907047"
00469     "3D67A8FCEBA7F21BBA932168CEC2A754247CA324081900636D87D001A605C4E5C31AEAA44A"
00470     "8DB9693617566726C7120AD15D96D917F5ADD0D86A5A9CDD7527644EED58A9BED014D9F4C0"
00471     "F62FCAD8D7D5DA2899D57C5510EF15B7482CB3E087493959AA12D73D55E194E48CD128FBA2"
00472     "788829AB4A86C325D60227074EA55BEB601128D5FF42371E74972B802D3117E1F2AF201817"
00473     "7D4E994903996DC365D3B52582AF2085EE5919DA7E84A2F3A200F4F6E76F11CC95F87BD2EE"
00474     "5935CDA0D52D9EE6FBA6D59E80427EEF4B80372091C65E059B566C38A6B3664974396BD038"
00475     "3A35EED0494C2BDDC167725C4160AFA58D408978D5A345EB03A55127C6BFB08FC0B4EF7037"
00476     "482EA724D9D121E26F2249C0C7A927946A846A67FB7C565E86C8C59A49265ECC4927A8FC88"
00477     "873DDEAEF6F8B629BAA1819807F38AFAC501EC9F206346581174A037B1D7EA56628C660E17"
00478     "BA0A97F2E189299A7844FEDC562BE678B0606CE18493F717559365F66201AA598C68F99A8F"
00479     "28AFD74AAB0E3619BC342AE9F36D3811EEB52C8DF6544C96B679CF434B9C64FCA0F43DA73B"
00480     "9BC8FEBD609678F5E691AA706E3586176ACDC25652BCB68D98417D3FEE12B23724EBDF496D"
00481     "6AC3B35EF706EAC11BB0090066F3C421C3E298B00821C90DDADE472331B9B2CA1F1E53E6D6"
00482     "8BFB059EE0811EB7E7FDBEF394E3E35E59BCC4B63FA8E09851843F28CD941F9A97E5B72C8D"
00483     "CB9D32F37B79DFDD1B9D7C22C10E6F65DF07DBC7734430B811BCDD450A10F491B1E0DA3746"
00484     "868622E457774D0BCB1B29F9C1CB161DEFAE78354B7F718D18F07ED6672665D35562D6F05A"
00485     "2B3666B667A7C57D2617711F27DEC15D0A4079519D0ACC6B3591EBF0A61A12C42A6820CBA7"
00486     "55C07FC531610C4192A0BE9BF241BBC5ADF758FAA4DE47C6B766AFAD728BB1D2C672EE0969"
00487     "303E42AF789FB295E766E90319D98F0618C1B4B8BD31CD2C5EBABEADA30C6E43D03F380F4C"
00488     "2B9C3AEA20BC89B852BFCD042AD3C9440AB395E9E251DA758D317E39792973083B20B79BA3"
00489     "6A14057E1DFFADCF903B10F8D3D4D35222E4584D520D6FF675D3EBE9FB43ED1B79252648A5"
00490     "E0C2552E388AA3D1168790A09854F1151A94DB4E5AAF1C941FA61B196D869B16051B75F56A"
00491     "C1BC769C6E7A7EC173264A2E56BDE4BB9952A422855D65AF1C209D561F8210FFD7C2FD6A54"
00492     "0BE788BE3D2822D49DBD2C6B33299604A835F07A34",  // PRN 12
00493     "0C0184CC1854C2013817663EBD9E23F3B78DB8009E761F1B38C2DA2036F26819D2DC68AC38"
00494     "C18FC15EB00BDBF5A11316C7B6075BF9DEAE9B0A651527DB3E3B3039BA01398B3923A2C40D"
00495     "C4706A0BC186D2F8FFB124EDE35E8C427FB6FF05A3C2ECB81E275D1AE370AFDFB6F731E27F"
00496     "4B22B877B1990FE479239526C655796BBB8545051527ED0E532E0440048912C64D0913227C"
00497     "2F6657B8B80FDA076E49B7E3DBA1D0E23C9D7F43280A324DF16BEE2A87B87C21692FEACBC3"
00498     "348C9186DC6C0BCB7C1DBEC60AFCF4F1D76FD3D45A24ACFAEB8E7C95145D07427CD6700301"
00499     "9AB57DC233D566BB9A20AB82ED4130D9B4C0FAA6F2EFEDC8977498315FED436F67F3B17087"
00500     "7040265EBA49415B1B2904C8277A11598432B973AA63F628456AF2320D3478FE669E2835C0"
00501     "0D13EEC7B6ABC6BB94F1677C9A05B2D0CB87ED3E529AE1D2E3354B1E30C6634E10AB3BB0F3"
00502     "9815049DD5321B11B80CF1D1F1426A413D0D11FAE969F3BA266F4F7B60684224A6279380D0"
00503     "8D46FFBA2E1D684416405A02A2AB2E260ADB23498AF716EE58B03C8CC442DE0EA201C4DE39"
00504     "FA7038EA4E0E1544BB08F149F57F03FA910E09C27303038EDCA21A618A54DA87CA747763EE"
00505     "6D4D1DD4912AB7E3BBFC999939F82563B944CFC5F4460D41A47573DA92CBB28439BCDDBEBF"
00506     "F16491C404B356E5889D33A034680F8648F2CBB1888FEAFC410FD5A3C8ED85B90EC875D7D4"
00507     "466904F5D4A4048D8DAD848313232B80FEBAA09D4D86F70045C0C2499A93610DD088A6263D"
00508     "E5AD71067B9B3C82E20658AE98677889BDFEC65C2574E4AB7C9059F1C13CBEC17EC630F9E5"
00509     "B891DCDE1E81552738D4C8612F3BEBD570EB39B24EC94A153465D1326765D9B8F23D7E4AE3"
00510     "F2782F6F9738C189D0D20C4DFCCCD37D9A510250F0941423CB9055C7BFED84520AB119FF92"
00511     "1CF81CBC6967E9CE1D759E5F4481ACCC5881F9328093B63ACB7D04404D8C0FAE8747D39A27"
00512     "0399000CB690CDE17A46E81DC5CCED183BB863FD7A535CCBC592C25B059496C45DAD179B83"
00513     "EE02543B5EF93213E10E2BDD480F7C7D007CA2988942F65E46B886EBCEF05498C655048C4B"
00514     "32A081A059A28DCF4D84AAA586F0291F322B25D186FCB5ED2DE7AF7EBE58CD8314CF846AA3"
00515     "9C67E2F284162524A14BD2765642B31F31EF4704548C13D5F5A132E983203EEB91795BCA93"
00516     "4BA7EB26DAA5450745333446D2B86ADE99788476A3417DB7FE9E53F4906CAECBBA98465796"
00517     "9B4592F9213612AE6812C215D5C6E28BC89BB76B9CCC937BA4D4C6368CD4C491BDF41B242B"
00518     "D8FCF9DE2D35907F08AB80BE71EF90F84BFDBC6777A020DC97BCA1E963050FF991A4F0E1E4"
00519     "68813AE1CCF05B8190123773344AD6B5A41A87A7D7C4F0AAD241FF7D3AFFFF40D7E6D5E62F9"
00520     "884B63E23C3A8B96260A6E9E3BCEC001D487E3E75DF4D35959633BE719FE592C4659970C0B"
00521     "67596E9CDA4EE3B05237C02CF2E36FFDB7CAE37F0BE993FAC46DA6F56B15FA74EB4511AEC2"
00522     "C861C2A9253259268A9BB95468FB02867394AA8D83EEB6464A65CD15DAB6A696FABAC4661F"
00523     "439F377560F3D2F3B668A7648BA4017DB5083BDD0D76B0DE99FBC06D7938D7D2F0566D3DDD"
00524     "036CF6A5E336359F0896B9B1E3B685D963A0EA70CAC175574C5AD41E20EA1332CF103BDE6B"
00525     "1C3B3F117C21E8FE7450866C5036C244856B5FAC176671270EF7225BE7CC36D24498C92E53"
00526     "06C2870FA7839DB1D7646A62AE0623EEF3199E8DBE32020B71F23FEF2F2BF3C9E0EBF0FDC6"
00527     "29D97837B97D76C99398EAB044AFEEBD5BFBC35FD4",  // PRN 13
00528     "8767E0FD01FC0DFEFA6640DA0B47AB5B2B2CFD25C8943FDAE6AF46C3898AA1FD2E2DD0FE19"
00529     "D1A9BD0353A16ED03C448B7005D56D108FF2035E948E3C4FB8F0F928A5797D491389AC431F"
00530     "271DD1C8FDDA2B1FE623D410190FA81F74E042C0BE9F97F99120DDF442D5A4ECD1EC9789DB"
00531     "DFA8C8E577E0AC5A1BA7DD6D5A474CAC3C2C04E71A14DD2244A30D64B17E4C9B081EA8AF01"
00532     "75D7D0A6E039B6A289EFEEAA20763B1F106667745EE2795CC588721BFBCA6D659D81192A02"
00533     "FF2CE11CC878CDCE73CAC9B92A972CC9340ABA7A03C3384ECD28301103F347D35FCC9B12B8"
00534     "5FFF170894F7D104F34D95B56A23C486CD9E7520EF8CC0BD928EACCFDC5D230D77EB9B19EE"
00535     "274113E1F8BFA79FCA93EF54B86900859BA6688FD29444E03E2F14033F076FF2F237111290"
00536     "99051C59E0126A907B57905A021F024D98736C3F5D1AA11C25F32319546F91DCFD4940FDC7"
00537     "C4428CE78A998694E6B19AA3CFF470AE4582666D350ACB02CFECD0E9EC39D1607912B89247"
00538     "9EF94CC67797F443EE6BA4008F46C0E61C3819C5F82DE7A7802E62E7050C8AC506186878F0"
00539     "8ADCA8A7674DC3D0D156060AB7BB08511D2F8DB46EAB9B46AF359135030A9EB46A6137A941"
```

```
00540     "9D97C7069147BF5065712B1F03BB8B328CBD1E583F59B662B64C0F06ECEC8B71AA2213EC29"
00541     "F8968E0F3093AD04E739F01DF51C644FD4CB65AEFC7073018DDAA81185627B613242260CA7"
00542     "A9270FDBAC3EBEF62865D0341B5392DD5FEA4827D60D5AD1C9AF0739CF5BFE0F658E72C436"
00543     "BFABEFCC403B87AD4B64F3143DBAE1D2CC25130E65D72ADECA1A8FDE09EDFD723072D53879"
00544     "2F50A067BA78D7E4749F0344AB1AC1CC0E8EFA1A42515877C4539EDE35556D444516474628"
00545     "BE9D8B29DE428BEFA12EF99C985F1EED4F937BA4794776C63033C03EE356991607473835A5"
00546     "0718E473D036F3C522A9DE2E49FEC2E686F17A5023FB7E82E94B26A4AC6236E0B656A938BD"
00547     "7A8575D734F53CA419836110DF78197D7DE4D45B6589CC31F5057759B8402F43ED9A423A7D"
00548     "69DB02EB9357B1E51207161E08FFEC8FE286722159C570ED4AF63EB554BBA582404C0E978C"
00549     "EAEC1CC1404CD15BA22EE6D8C98C4278C1714695927C2389D74BBAC72C2516AF1E5D0C5206"
00550     "6A656170091B68A7E4C8399ADE63605E9DA1727C189C541A404B7335DA27843A32E470FB4D"
00551     "ED8EB5EFCFD30BF15AAF37D7CFF4B545DD09DE3E7C8D0E77427B3C0764608912616E5DA2A3"
00552     "B885688F25E83DC8D632BC00015C2C8E6923A3D9E7C7344CDE87D6F0DEECA2006F644F7BDF"
00553     "A0A524759B2B16313501A1CB8D4D762D3E9D9C8D32154F69F0DBABEBD26AEDA947485FE0B4"
00554     "0D1D7731D7F4E41D1EA5FC645DE31CD850273B6EF9C374CC8E8149960B349BA16FBDFAADFF"
00555     "12E86EF05AA38B6B17EEF7B55D27D7CE27FEC916DF99F0715F1DDFB93C7E1130EDFA44663C"
00556     "F19F4008D5CACA48262C1E8209D0C353ACD72A542B93EF18EC7E15FD4C9B1633161D94B285"
00557     "FE8CFAC30FBB1E140E9561A35715AF982370AB4DDF25BC30850E5F0509297289C0045BD3B3"
00558     "912843A9B0BB88CB7E961639849B8A0A75E3E903182F0CC6C7ECAD4A38081BEB18DA72EF9F"
00559     "9439DA7DB2B4D59AF6CB71808E604E56E1D4BFEF627226FC28875888BD486D0C8052068D12"
00560     "BBEC6B27E9DCDD73DBC93D33EE66AB63936C2273525A517280FB4BC33A724D20BB7D988B93"
00561     "8018EAC4738D03D583A71C16671B87CD60BD6FDD101BF2AD07492A794FB8696E84CF24B970"
00562     "7D7706F0AEED606C3A7FB5B1DF7FCF86E65F6DEAA4",  // PRN 14
00563     "CB8EFFE09FE09C622B34574E74D4B061C284F96FB7937481FA7C204F449BE9B2DB0957E428"
00564     "D177C5525C67BD75D75989EE98D97C4D0DEB5E72E2B0DD62E66EDBEF0D22D2EA97D42A3374"
00565     "B9B659804F1EEB01D7743DD9AEADDEF356C7C4FE94077411DE58DFA5035271015AF65FD1D7"
00566     "8C0F17632C10EA02C537706A7C14E3A1D7124FCC11F579333A68A1AF70BD3E86F7B1D13A3B"
00567     "4EBCDE343F36A2B226DC53A0E1A084EE966272EE852A349D7A5606F1AFDBD81B8C3CB556DA"
00568     "E83D6A34ED4F9C06BBFFCD5953C51814FE0F4CF5A697EEF5559886D92F811004C04FB399E0"
00569     "6DFA845DEF183160BEF8546F6E8995E403FE0E4640086463 30CEC4B4149DC464F229A47E0C"
00570     "D9F89C83A28FBD0C1E758D380C2A7C17A487191A46C04C23F7906418B69E2527F329358895"
00571     "2919D220EDC36BD9687C7347AEEEFC8CD24E91299900AE4ACF018FA2829E215CB13CDDECC9"
00572     "18DC85BBFA317FEF7D2FFA9E3A30D9D1F1B9DFA0AD0D641E800850B43D7E0B187F2ACFA74F"
00573     "C5DBC99471A7511B838BEBD6E2744C3AA9AA9D46060C83FFF803C148C35209F840AC23D60F"
00574     "D046B461CC2D776AF51612DD166E1450B8A998052EBB25D1C8F426391A8571C88C2012A894"
00575     "0CB0B4DB3223CCB39021D20142FB5039A6ACB250994AD7A99B1A43494EFFBD8D3692F213B4"
00576     "6C6CE9AF3477BD6061398CB19C099F27F48F4F785E9E741988412F070F39727FB8B74071E0"
00577     "07210D939B0EC5D6148A5931D38A030374E5C0B575AE27D49DF49A77245BE310E49262C56A"
00578     "616C757A80C5989F27962303825EAA3AA014A4C9C9E5B3CF93F1A031F82D6A0C877277EC41"
00579     "356D1BE2154F03D807940A1CB2848A6A23B905BDE9A907F7C20E3335863890141267 65CBA8"
00580     "D5C8951CE09DF085B5F18A24E1E5D638651687CD2A2F75CBAF738F65A8B24E7C67E64FFD9C"
00581     "B8A34A473F11996E3663607A49EF663C4D87A0F65D92067132B5EF3CFCFA9B49FF66CB6E36"
00582     "3BC1C7EAAC392605200A5EE704D1E754CBD1C9CDAB75355BCA41DC129722F855215220092C"
00583     "F6248609DE9EDC5603562F5C82B5414BA305CEFA60020E1CD93FC756241FE7A20CB4B550D9"
00584     "80A7266E7B23B6FFCFFC4461F925A14EF6584B02005E1D795ECFEF93312B3369C52A8A2ABF"
00585     "99AB5255E8E09C05AC722F37A5E95AE46B226C7319411750562DBD0296F0CD1326055577BE"
00586     "AAF270E75F4433D5312038854C30AB23BD51F1246EC46599FEC1B628137A73A702606112CF"
00587     "61CEA7C3977BCA6701C8E5F63132C15EFF01CA1EDFA5CCA8FC0E3FAD392F21972F857B53F6"
00588     "2CAF3BDB75D556E6857067015E11C5E217F5D1C1AB1D603571F9EDDE04499E23D39DCAAEF1"
00589     "6263A5EBAB9BEAD8F61B0ED942EF261E0ACD309A903E73AC4E4569A65FE6490F6DD6842812"
00590     "DF61CD858D9AB54D3456C89CFEDF2621426E3F9D13FB94177104F6EB4E30B018C01F78CB4F"
00591     "A9C89F453B87FA7D4716B96B0BD2DC532D4AA7D8FB9D264B095BCE39BA017E8BB90D600DD2"
00592     "C58647B9D95F63783B4B90BDB83A175E5201784AE000A493A387E16F5F0C5CAEBF62A360A6"
00593     "A927B3FE4CBFD310A3163BFFF1836FBF9770583FA4F88E5A6E7BE5796145890A41FAE8BD18"
00594     "39110EC05C325012E19C69C3868C7507D04458C074DB3440A0E018A23CDD32ED35E7A28DC8"
00595     "AA8BD63C8CD0A9F4876B56C04D5D3D5CE52A0601ABE2AD3CF3106CFEA6D761335F420EE3D2"
00596     "77DA358DF4F6813D543424653B256E325346F677EB7F69B2FAF9536DE80E8EB018D9784845"
00597     "09804D600783708D48C4F7100E0AF891BDFC75352C",  // PRN 15
00598     "93EBCDC22C36C3BDC8B579B31AD5F5C9B2B08218BCCD1B29829A365DF4A8D5259753381B67"
00599     "86877A5250DCE6A2B182734E363B39560D3DF44E88CD63943247650 87DD1A77B3D8121CBA6"
00600     "1F665F9B719E75B740BD930D6FEF316325B8B3A2FDD93AC488714AD9AB525248EF60B2649B"
00601     "70DDE6849BED793264062B38EE74270CC190C6A86AC4E106781BB3D09F0A6D7AA4DDBB0116"
00602     "D3452CE041BFF4247819FD8D78304088355D8DE741B1A6461C74A4832ACF4D4004BFB22F6C"
00603     "9BB397D7CADC6FC05AB74919CA346F9A54DBDA80F8E1303A1D43239C5DE043C2AC21E182CF"
00604     "49BEF9BCFC109BC884E41FE2D8669915701992200BCB938A745FB0994E6AD01187C4A24212"
00605     "DA421A178E42216287DB1F6A25E8370E4209E09057E16253BF477A061399AD1B3FE985D6A1"
00606     "0D8B05FD94B67F878649862D68F285EC0BFC06B7006589A590AE7B116EFEFC871010993F47"
00607     "68197473BF22AD54B2A8FDB0BBF2B181A2A1CB9FF0D0989F2F68B339A8D8B27FE1781F3DD6"
00608     "F7CB972D73E3DD702FCD0CC3CBDF66A6237460E3B2F45DE273D2BCADB96C118DCC19F7C1E7"
00609     "03BCBAA04DA5370A5CB3B78112B8E95B63328B43C51557AD84B3564749C2802B52EB334583"
00610     "94747C176E119E1774BC905119312963401C1DEE02964DC9F1D836069CFBB9B33791C908AD"
00611     "CC717B08FE3C0AE98C0D5DC0C22876595AF95A02AB33BCE3502A92E532E72D0E3455A3D7EE"
00612     "796B3CD5D61DB902B8C0057F2D47F2655E95DF55DA562F632A4F9A2CA8B1D2DB8F5745A713"
00613     "18C37043D3EB08F9950D06C1301F9E9FAF4EF560B238BEC483DF0E017E3CB27A251156221F"
00614     "183F3E9B0581838D04BEB5DFD9EFC26B979BCAC7DF631F1ECCB777E2617EE7386C8BE9834D"
00615     "26B96D80074BBEFE8CBBD203299AFAE8E6F90273B92182CE7EDCA9154E0A33AF9981B8CEC9"
00616     "4E6A7C78C4716B47790EAFC771C3DFDE276E4E617CC0DA50F05099B9BF587127D1D18D5F89"
00617     "2E0090C8D033F21E7382738F1EAACFBA493D776821DBDFCCBC24B1D5F877B157AE6BF6D307"
00618     "AE9BA54CD053E68594505D038747A7D8450EC3CB4F2E980C5D97A2AFB31147906CFD3E9000"
00619     "BF166EACA8D16A532BCF69526FB4B86B1F71D2C5F38958EB11F9ACC438CCE14E88A11DC431"
00620     "CE4E20B99DF0ABDD2B5AF358260A64012F4A408C6840D365AB95167B54FD479E8527B5FF66"
00621     "2161F75FBA7CF790DBAA51613DE7C52A2A94E4AB78E051F32D10233161A4EBD8682389D091"
00622     "B5C4509171650ABA574F1FE954698268450E11D2FB52C1AB3E4DF426D26DC6D5319FB762CC"
00623     "CDD552F4B1BF4E82AD1F78606C644185AE87827F022C9508805472136DC48E71D1C4594441"
00624     "3485EC1CC72BC092D6F9990148E12C2A3FFB3D8C9F91C18CC9086F176733A83F2CC9BAFFBB"
00625     "D810A99B66DC3B399891B2045BA402749B55E7E363F5D062427A9BFA8B599367F1D07A38DA"
00626     "16074F75A4E095A528C26DE4083B044055F73D207A6571196263D8BF5A53B7679791BDE49E"
```

```
00627     "869F0F58431829396A34415CDF4CB986281840BC48857361281CB4BADF8FF5B86ADC76FD3A"
00628     "812D7641C5FC88A21F137A6980B42FD3858E2A057BD4D3FF57A5ED3F5C41C71C70036E1762"
00629     "C0079DEDE282175072A073A90AB83C9DB447AC07E6BF99DB04BF276FD876F4E4816B029433"
00630     "3BF3403DACA2077BCAC3889B55E76D62F58DE7D1C49816D03DA2D492D1F3CDB2B6B186439C"
00631     "BBFAABB645E9745F404538BE8FC4C698765A3C67D01CB54D404D2A2E78098FC334F0A375EB"
00632     "41C3031F57FF9A773612445AD3D98BD09C949616D4",  // PRN 16
00633     "5D55CEF58C74B4C681E813564F1903FDB56BE053280D438BBE5E2480CF279CE47FBAFD179E"
00634     "270D838F764C2E7F496AEA9C0800056F2E0456EAA7E59CB20432B65875455975AB37186D6B"
00635     "06FC7D0BEE743CB974DF92B19F4D2EEE82F36598F8E400DCAACA4EEB110ABE352D3A0F58DE"
00636     "FF8D51DBDCE5C23E84D1166FFEE5428541ACA407B9578E4295C3F79F9376384B630D25F112"
00637     "71C6234B9AA9C1B4C8015AC2C73CF768B9EC30C61BBFEAE1426F72FFBC995A834C9D4FB835"
00638     "0C301133F123313370C33FCFD711D1847CFA84E782F4B39762A02F9476D5F45237A29E3E26"
00639     "9D16DAD60033E0907D5431C1CE36D37BD388024A020B612E1DE42492A3759AB82AB393237F"
00640     "4A1AFEB8217D2733916B7B610342C0D182C17BFBD4556B071D82E6C6E554F2127654C15671"
00641     "FA1A843347BDFBF9498C7B5E7636318C3EDBAF7BC4BCE123698A05538DD5D42F01970A41A7"
00642     "5F0A46E61520911D86A39D892700AA23A033EDC836EBC17CD2EA274252F53A4BFFD5CAD88F"
00643     "8066744C737B364A0DD0B5FFD96787F7D18357DF728FB66B212DD4DACAF7218292216808E7"
00644     "B81E3514BDA947E436A301C4181CE53A298250835AB98DC89D0A3D6240C57C699A5BC31EC4"
00645     "CBF8C111149CF6802942256520BC7FA22968556E2B390E63C9DB7D0AB79939924ABF068BC3"
00646     "D44264D607F088A38FA67F8E25933091BD3EB0A3590F3E62E63CC6C66AF66128F7C3F97B76"
00647     "E62F1AB82F24D8C147FB2C8BE207FBAD7E5CD355230EFD48DC9FDAA4FD1C970781674E901D"
00648     "35D7C651EEFEE2356F825ACD348FF0FF590E8CBFA0332F77A063581E6F19F53CDF1BFE9525"
00649     "85E93EA13A83BBAD06070850EC824397A9FF13227F4C60D65B67AF810EF1BF5D5EA95A5512"
00650     "F95AECAEFA25BFBB24CBE6785E91A4874E81E465AB91B1F2E990F37AB145645F2681B47592"
00651     "990CA78264DF8902620E796D4DA9A260D1BA67DF9BB0292561FACCB2478AD7F7D7CAB3DF17"
00652     "9897596604F2B0F1E3F10CF842C311F49C6B5EFD5D73DFEFF59127384D3B4D855F8C412E94"
00653     "9E3A941782C186413946DF8AE33C9640704CB0D930C12F5FA1C4CEC5883CDA699E29401956"
00654     "2948AB028548FA974D6C56AD667023F9C417F067810C4C7E4A712FC3417A92C075423A57C2"
00655     "08AF683DDD695AAAA5F027CE535412DFE0029B292D32B2366DF978294FD263DAF8C6A6F7CC"
00656     "895BEB1437CA189C2AD9B6FD5BAC06DDC3D075DF55BAA6E75865CE375DBAE862EEB7A51D0D"
00657     "DB5DFE03234CC8F14EA99110DE78A3463546C6CCE5F7C623028134968C59E247E642CE17D8"
00658     "933B6701DB7952FBCDC6CAF7F2FD24A241D82D8BB1C0F5D55BF55E57BE4AFEB4200C1C5793"
00659     "8F4A6B1BE9AA02FA86AA811577BCF1D6BE545D6F12046735AE963DC7FDAA9A2F7270B8BB37"
00660     "B3BCEF7C73F59013041B6CF9FE10FAAA86DE369E868FEE7A37F62E2E9CB1409DFF11FBB09A"
00661     "76AEEAD0F8E443582319F6F74ED0AA3229BB7A50043636E881BD139EB7442FA35F5DE1D6CC"
00662     "227B921A1343E34030D2F1F361D7651C487282C907577765161DB781C49655A5F87BBCB68D"
00663     "08C04290B4A2A94B61BD0C7CFA41821B7F5641B899E90DA026290FDCFA4FB956414523041E"
00664     "14ABD7B1AC7CDAA1148A03B6EA17C49503F01707BD3373EACFD4E248BA0F8D4B845251BAA3"
00665     "5D41DB186E10464F6FCCD2CAFC9CB06FFE9DC97FF41E07084B00B7B31FADE0E093C8968D1C"
00666     "EEF9810986821F1741F11A71F5F20C809F54852307FBF33A6C33C6A230EC7CB2CC6C44A3A5"
00667     "3AFA49477037E55BE691ABDF6D4A852799453930A4",  // PRN 17
00668     "B19B7C4DE003906A0001E0DEEF57CFDD2B50514CDD6F57FAA2859061678DB9F8B5AAC67442"
00669     "D8B737F635DC297E3A46C34735AE4D2C607B06964D1FE264B7786B8B0604781D4814E2A81F"
00670     "3A0250D4EDDB77C121B962D0E7109A738630557F5AD47F95A0971A58704CB5B53AC378781B"
00671     "48D67FE5D5349440261C58C26D9370DF37255CFE845F7B0F735CAACF4921CC9D55B3431C6C"
00672     "888787D7C2B436C9DCF1DB94FF0D510BCF15DF0DE42AAC7640741B4893C960FADDCE4261E6"
00673     "284681BDBFB0F471C864FD560DFD87DDFEB4EE7A1267E285BBF2966F6A8BB408A105034220"
00674     "F2565E38F6A15A45B7FFA82DA63965BF44A182FB4672E90BA7933A0120A449323340D0368C"
00675     "F057DF299A8CF60035170D197104B394CA9F2A05983980A1BD93CF3AF8799E613BCEF26234"
00676     "DB2550A15BD53FFA306D3F616D78A6204AD1D50FF4F9972B0CFEB48B76516155791C14E0C9"
00677     "E18457900CE13F672882BF423400D2E6FF94CB16BF3071F44CF2DFE8317A9B761FDC8FDE90"
00678     "D7D05827521B070492AEAC90187EE48D93374B463C3381C7D80178F54C676479BC92470FEA"
00679     "7069CBA95BFDAF099A994AB8244CA786E7767B8400464BD08D5C560D61F27A96E3886A4E82"
00680     "1E0087E7A56F96DC6F6E85F1FDFB5C82FD7BD48B63EA794BDE01439FE4C3D7F08864263A0A"
00681     "8B1C1ADAA5DDE86B99EB2A619A12A1A76A39D8B1F04AB55B67D15C7878651BA1F662165875"
00682     "7B890DB0D2BB533BA1C6A5E3226DB4FEE7D561842EFCB5B459598CF5AD3F98217CA720B678"
00683     "1E83C2297C96F4F51CBBBEFA708AD4B3F682FA0C67C82A5F5797030E0724AE9715A911CEE3"
00684     "47E80F774EC780DF7E2279924F67ADF9D07FE4F56DE99E06B58099E1104DCE62CCA82D100D"
00685     "90BAF44F26F19ED7AEDEAE609C04C530CFBEBDD161A2C324A382FC8234D405BF900F0B6CE8"
00686     "95F9BD534A6474680A7D63C0CF7B4AF1D1E93958CFC02FB379AC09B205708A68D11683B928"
00687     "2564C06290F5E6B57E3E7A75D30C9E21DF8F27F85E1FF347B32105A000BC613006BD7FA6D8"
00688     "45C8E6D4053FF012D0D85F99737D045871BE017D672C8FDEB16F944E3020BA7FF0351A303A"
00689     "0D3280D71CBDE42F55657671A79CA30A7A33E744E74B34D27453654DCC8E87911B3701FBC2"
00690     "7CD16C4D9D6C29D1BB9AFCCDAB28DB6C2B42C5D6BC1D1386AB26669985B626BD54791930CE"
00691     "D4A13716BC9AF8F4DC37A04184FCF711EFF515B92C943E53879B5098BCC4E87B98000BD0BA"
00692     "2E1CA3C7A97AC95C1AD145EBA3950B154888272A0ED49289157091DB2C55D43AE77FA3384B"
00693     "CE92E5840459B405237E194B6F5788F16B217D8BF7539EF8809FC7AE1BA931DBDB2A7D15F1"
00694     "39EF4FDA60F02FFF9462777647D675309AAB8201B5FCC1ED7AE066FC0E4F32D260DD9406BC"
00695     "EFE7DC5DFB38D66CFA08E5CEDBA636EFC23B3CB7AC38FF7C390542C9814A86D2D8F676720B"
00696     "7E2FD93217F916F10FF34A1AF34E3B8563319E92A3BAEE34258B4DD63FBFED854E08B1A2F4"
00697     "B560FFC40AA1048F081E66748E05EC3EA02837272FBD9CBF1A658B86B19B7DFC63436185D0"
00698     "2C6411142291645692A92E3EA6A113B5436B0BB78990942146956795DFE003A4AB6D9ED875"
00699     "FE7E5B895EBD69C43BB7317CA5C97BCAEA97CFC6068431B24673FEA931EC1DC1A3F72625AD"
00700     "531CD89CBEB0E07F47B26971059FEEAE3986E23286D13BADEA3585D73F1092E41AD041D591"
00701     "20D3D25FD3BB0AAABABA738049C91B22649E715F3710EA51C41C1705C4F1FFB1DCEC85F0EE"
00702     "704240EA8A20526DC01C7C8389B21DA979B99FBF44",  // PRN 18
00703     "5805FC5E7042027201F3517AB8BA6F5C3BFB0BC7657337ECCC99CA480BA9C3D36033336AFA"
00704     "C0BE172AF430A98D8462D46453E935805E594C18C0EBF735D90D5027067006193854E7DB64"
00705     "24688BD715A7BE72EAF93C2F763F3BC340EF4BCF90B03AF20C28A59F72114F693848289873"
00706     "BE3869590A64658C0A3938A414BDE86BE8DBC25165217B1605EB0402FED48DC09139751DC1"
00707     "DCFF532CBF599E9AA13F3A72142E05A1D8E47A4044269A0B414976670094C9785811628718"
00708     "C6FE5201221A5822E29DAD564B6EECF025DE7783B7842DEA40D4F72D083D112C608C596F25"
00709     "E09EA85BC6CBDEA39300774301C1C36790D7D9C0B59741F6B53E16BCABF8120112C81D70F3"
00710     "0E44561EBF3A3A4571F207305BB3A292C46C46E285C246EE9A7442F869711F52D921EF379D"
00711     "F7CFEC4CE64BAE4D7AE7DBA3CAF47EFCEEE78B477C02550D1DABE4FF4EC67A38A705F60706"
00712     "D948C017E333B645C0D80AAC1CEA393CF3ECF711CA3D59ADD7CA02DB6C5E3F259B4ABC138D"
00713     "24C47A650EFB96AAAE93E69F08077D0CF38B230F199C3B353229C529278DB1F5846086BC8F"
```

```
00714     "5F54F9631423105673FE92799FFBE5DE6ED167658BA5DB5AAAF921926FFC27956758CAE8B2"
00715     "367E3C5C5C6CC81DA875AEBC8F3460275C4EE6EF43570B4F571994C7690DFDA3DE311F2FB7"
00716     "7057732B19E6FFC96221DE1DAAF9B8CCC21609160457365877519D2B332FA3A85B9801EF43"
00717     "5D04DAEC3643CBB5413A036CC1AA0968A40F4EB5A76D2D0C3DFCC53C97BB3896E855957B19"
00718     "41FEF59048562171F6241AF278F6D6B580B4C94C5CED65CBC0F810D46E864C2B567BEC920F"
00719     "6D1DE42CDF32300805E6022EF184B43C1733F809F98B129C937930B984F5539F574AE144ED"
00720     "AF7BABBF379EF1A1AE69B7B4FF9212BA817A4F223FA8B182FEA231E7029F80DB126641725C"
00721     "36AB4AE950725435561C6FE37F960E37927585026B8ADD92E4ACA3B4500F0CBE078680D16D"
00722     "BB90ADF6A127FBC7F791B4F93D5463F8C71EEC1EBDEECA881FBDF3D2940C2EA0F8E4DEA107"
00723     "4EB5C0C9EB34534C9C3992E988DAF8979FD8AA90F540F028708BC6D0EBBFC00FA946DEE5E9"
00724     "D01CC272A4FE61E5B86FF1347340FE64F650996F65E5702A61FB53EADCD457A5ED66DCFE1B"
00725     "019A79104842C51B60142267392DC51F86F69367C25B999B3153C1B7895298B6B1B43EFCD8"
00726     "CEC3694F90BBF1A417C48E5CC4983782F874A7C64E05BBF462220950DA9E8F56BA324002E9"
00727     "1D934469225E57309C7F562407DB900267553B85C4314A63351F20D57DED0B3D210269B09B"
00728     "976C07878C324F74758D9D507A4CCEB4AA30CC9CD05CEF32F1174C7B0F7AF93875F66E3A7B"
00729     "12F163920825885CBE5092CB7728A1B74B2F3FA9A01264214761DC5101BB0C18D65BF972A3"
00730     "587F95B3E13F52EB6D1F334B15220ADA5A59497558EAA0F4ED358A995F7444B633FEB900EF"
00731     "D9528B51FD82F5AA30699900D3EA3F3644E4FE3F6F2900ED45DFD997FA905D91521CD9C898"
00732     "AF4CE09B6EE00A15057C17DF4C656E00DD876529CD75A169977962DF3F6E4CB6950BCA37A7"
00733     "9C98F108AA7B02AA488AA4ECE05F702B468C20C81630F51F884AD74A76491DA378DDB8290E"
00734     "CB8F5879F019BFB416BAF1D9361126F3514557403415F286C8E77C0BCE4FDF0F4EAA96D19F"
00735     "22F189971155F749EC19687657B245D392BDDA62935A44BF6C82E0AF3593BE06745099F06D"
00736     "C4A66C0768F17916AA360988D5E578D169CA84AD1A36B12CA4929223BCC52AA9B47FB138E9"
00737     "99CD6423D1171AD3BF87B5D405A174B0398A689DB4",  // PRN 19
00738     "F99EA1599DA410098772B05B5D7F9AFD485A04CA036BAA9C71AD5DAA3EA5F1C757BC83F3C0"
00739     "028DCB93DDEFFD157EFC2CB29C0F8146D6704D34091EAF1EAA29AB915E652309491F0BDE63"
00740     "2172914C132CB9A533E97F89F567A7A9B0776B400FDBEE71B49AAAEAD156753F5BF564CACE"
00741     "2C1285C4E65B5D8B2803A11B22350ADF79645AB6E458B1C9E2031C4D4D6361F77B53329C35"
00742     "30F6EF727225DE668A960FEE1F6D7588BC126D0E2AD23C2ACDED06AA2331E9D0E7F99234A3"
00743     "91926F0DBE610C600964E89BE1842CB1F1551D5CDD4AE3FE45B628E9A990C872622C9257BE"
00744     "40A8A124FFD6B135BDD2EBAC05E2DDE5B7DCC13C76C27C8A3E778063F3DBB3488CC52836E9"
00745     "2A89641F4559F377273FD51FC45550C727689CA4D5CEBB26263F4268502658F40D4EC2BBC0"
00746     "DFD0E41F16A399F02711BCFA96C2456EB8A77FBEB852726B0DE4F927C13CCB329F8B9BDD76"
00747     "0D1DD299863BABF42EA7D7F6204BF13F7355610527A38E89F66D3D73522B421AF013197D64"
00748     "9D64EAD17F6A6BC8148D9989E786AAA94FA19077292015EF7CABA4B1545CA9CA20E5A816D6"
00749     "EFF7FCA31FDFABC611253FEE80FF0D75518001C2DD6C604387C2656E1362A3EFC1AB85461A"
00750     "5B55C6374733FB6B17090E6EC1186EDCE2C49EAFE84B139C66A2E8308CF2370095B0129B6C"
00751     "E7FF6E17D14F79258BB3E555EDDEF12E634A5AF748BA40735B90BA1E7DE7EA00D2F4A8B3BE"
00752     "D32BFAC17D10050B420DA32EB6D14035CEEA1B7951592AD75FB751FCCF190452FA88AEF450"
00753     "80161C5544AF80FB0FF93C62B02B2E5D1966F77760D446F2625C3FFC91A07A5E852BE406F1"
00754     "638B521579FF299227769DC4B88A805E168F08ABE606B5D745433584BEED712453785B1C01"
00755     "D2B9F1020BDB7EF11936F0BC449EF26DBC2D5FB35A2131BE9C0DE2746DDB6D0356294D0E09"
00756     "076C9A77A076F6B659B7B4F54A06BE2F353A3A0EDC6A5123598DF9BA1CA852A5D1FA16877A"
00757     "78D88F0B367E6C4471758F0663A7498CB1C8F13B03F0E2A941DA89C717A15143F4ECF594B0"
00758     "DE2E43B657DD87E4BF536ECAF3820127B1FBBCC0BF2E53B9E0E1CF10191BA9B43D4D33FAA3"
00759     "79B1B543C593F1BD5CC05E0B7511E8D4925BBFC5923AB0E2FA90D05ED580F56071A62A3344"
00760     "08D3C454C892328AE1DAF16CBD530ADA7B858DA3763BACBC989F056EE1D46192CFF6DDD7F0"
00761     "66D2EEDCEE5BC6B72551245EF8D10389984D3B3455A7B9664475CF6B837D72D42E49BA2F61"
00762     "298DE9BCC707209D7F1E56BCECE1B21FDE69293D2560B8940223E0AE966459C249751A2BAB"
00763     "98323F184532F4D6FA396D5C0AA2A4A41C75C638DC3DA5822AD7D3D7D51FE0811ED3673D87"
00764     "5039D3625362D7D00356E8461FBFFC49FE1B0624F6D30D550FE1F5FE4211341B9D57D21023"
00765     "7902BB7C62514AD872C1CA87976809717817AECE1E224733079478A5DDF5F2A067DDE111A3"
00766     "FF911D0B511A6AEE16046039B229C00D6D10317F66E1202449DF20CF28BBAB4112948C65DD"
00767     "7342D09059EE2701B36E2ACAA6A9948F818B61B9A3155729EEA670D4AEDD6F0AC314DB12E0"
00768     "DA62FBA01D419F7C48B09BEED87F7292FB0315A597CCCB7B51AF776509818A73A0BED7A656"
00769     "AEBD7ACA26466057979 40CB737548E83653E204B6A7A5AFE2EF16EBBFD73484314C5699EC8"
00770     "D24E899BAA0E202F37E3C8D8B5F75ED34417E608EEE4B26B453C07DFC6E62FAC215B6122CB"
00771     "14A07739E86FCF816398DD49970AE8A216BDD8C4C478647A8D07EE8BBCE6631C2292047415"
00772     "4B3C83942C842A728EDCE7466565D9A6BB030CE684",  // PRN 20
00773     "B23CE558D0DE0D584768E83A066B9BF6A12C78D173AA3262866339B86EB8A8D5208159C587"
00774     "3CB29E9C4C0BE918CE4E8DA85C85E070CAC29850CDE48E2DC5E0609F419CFA46EF650B9022"
00775     "273AF14BD9EA28FA856B7BCB580D8949C1F732F87EB34BC1257205BCB6820FDF3BD1E7B9C9"
00776     "9E114B3C7607D01235E111E3AB1F30E8A9311F0F30B601C833FB0D8C4D76AC1B02F85E93D4"
00777     "C3D0205F735E20223410CCF56317B6A12D07F3BCD42D2023BA0676A00E7268733C41C8C7DE"
00778     "D2555FBD2BE09584B3063BE41BFE93340955788603F1D52190B35FFB8148CEC733CFC1BA49"
00779     "F609ECD3C4512A99308043F3C377EB65870535388376C2C748D18A748F5828B3362666C49F"
00780     "803828B4A019F26D6A9362726E32B650A85523484C4BB2ACFCABD16E754EECE0466D1B3701"
00781     "75C37C8CE9DE236FC58ADFAB4FAEAAA12E048BF40CE084467C8835A1E209D2CD67C5BF01E0"
00782     "D62CC78A12130F5EF2E3CFCDEDF86D489C4C5D9AB99484CDD469F2F8EE5E14BE43B9EACB92"
00783     "E72237FB54F9615627EEEB193EFF888AA0A908327C61C50D20C613C933D736F6C3D7F60969"
00784     "A5D3770275081365BA8A57DF600F10A1652167AE3E18AD3715532B7C4BAD97980DFB0C4F69"
00785     "D9137EC7B21BC72AF16F0BEC30B94A4A5C2F55074ABC6233CDF4D8D347F4FA942944D9B253"
00786     "9D0211B24E2C7FCB56BA877371AD7701ED087269DD643C4B62E4B40D3F6572C0730E465C5A"
00787     "DE50FDDD11BD30C07EFF35AA6218D0B73B22095D320448E0671E1CA59BFD52A517A50932A0"
00788     "6C4897BA6F7E7038A39C92C20F6DB18D6F6AFB3563D10BDF48BC194B79FEA6861F1CB97562"
00789     "81E969968B65161D3980C4CBD036467850EDE9397321913D57C1EBD7743A44A12E8C85F8D2"
00790     "D1EFE2A8754B5ED0E93EE9649C53F28129130DEEE66E58181787137B262E1EB3FDFEE0604E"
00791     "D1AFC798B198044DD5D0A3E87A754C7EBF0852953ACF3371BBFA0FD5979901252EFF69473F"
00792     "C5655709EA193DCA2A2BA63D706C84F49802684F80C576706FCF1BCEB2892B9179A72B7CD8"
00793     "86AF36C7D3ED09270A4077C481736BB69A333AB4BD864AEC1EFA944A591608A2469C85AA64"
00794     "31C47F657AA9C2043F373A69323E735B56BC8177E1E37AB03BAE93A7B304EA1C5C52B244FC"
00795     "265EBD674FA7F1647E29CED527FCBC93CA7A56A4E771D7D2E576C40B2EC736DDBB4FD74111"
00796     "68C5C568C446D1E454249E6F7529AD83215EDB62F57D25272F351CB2C40965B1DB65EF5835"
00797     "923EC2D2FC7D8AD56B67980A3DBB77E6C4415F54F3FC00A790DBCF39A2FB699A20FC634F1B"
00798     "1C177C37898A81D7693B21020C6B592E5A6359AE395A841B434B675F1C30A9CDD0ECF987AB"
00799     "C2406178EB2DE941393AB4F96BCD43626978AEA548173413EC9DEAE7B53B175AFAAC5A1E02"
00800     "D6EF7E27AC41268C1FD8DD83D1AC55ED45D087083C0D136048B7F61EADB94C88CBA5503000"
```

```
00801     "A90F91A39A8C9E2DED567F7FA0DF605AD7870189600B941B99240A15310E7B3FCCF5ABDDB5"
00802     "6D50CECC06B4303BA39EFCAB2F7E9F5CC8F542E96034145115AFE3788DE0A4985EE1929D49"
00803     "35EF975E760DF7FED8A257718DFDC5A0B4B87024C59DCC6FEF6F2715582023C3C80EAADBC7"
00804     "AA8712FFAF80CF5E159CF5418A1CB50845C97A7F19DF3CC5FA344562DFDD819DD736E9ECEB"
00805     "7BD57AF324256934CE6DB0CF295AA0ADC5931D339027CCCDE1441891067DEA40F69F6EE071"
00806     "87F9B78DDF6D59CE02EC3BA6DD68F06851B10AFE9672EF84020CD53ECA25B3D5E527405AAA"
00807     "DF60593A8080373A387111EA898499F096FF46FC58",  // PRN 21
00808     "8515E8CF42FC33A8CEECBD6E1B65BE307A5BFEA9CA59FC51A382C24CCCF012C38DFAD3FF3D"
00809     "CA3180934DD6AA52D828439F1052A959DE23BDDC3EA3F2E8CD70F7A765F77549BDE6CF8E44"
00810     "4762DE51CBA3431439D1F1E42F1773FCA2F90E48126A000E69C0964B1955F586CDE1895419"
00811     "083322FB4E6528F61768F8BE4CDCB37FE9D02B27193A0BD79D15F3CC9AC6DDDDC82B02201B"
00812     "EAB86D13EA560F7EB5328AC0B534C52BB9D266F291425EF1607A19AECB9AD681815EC4FD18"
00813     "88BF5449421D92E473C21BD44F2C419A0B3E0603EE3C5C82E56466C5A76B76F81A6C394BA6"
00814     "93991A5808334EFF1E996AA1B1394A4DAAE00995F02E9518053FC744BA391214DF2464DB9F"
00815     "01A5A53870173B3E8FB6B9B698A8DEE80745D7F71B70B811867D3513197B32115882348FC7"
00816     "04B2AF8BD6FB3BCCE30361C1198B84865162734719D88BB62B7431FA39D452E0A1B0A7E455"
00817     "34BF76B6AEC48DC25B7487FEB4B5FFA9C56CC7C83E39AC196987747D664943A600B106F0C3"
00818     "D6B2D0D4425073EAAF89AB33BF6AD76B151CAC32A670A2B03BA31FBDB48B949D3074CD9F56"
00819     "D39553708CB0EF2BC96170E2FCAA3CB2811EF1C67FEF6FE85CD450FEA37A04A07EA732A696"
00820     "EAE8E76D497F5C350CDF72F8970FAD94C03909F67FC895785D5DB6C3A68641D68895AC8544"
00821     "C6A1096DC4CCD45A68BFED7ED86961B5E035E61E9C126319A7B3C1468457C423C4A4E2DD1D"
00822     "42BB90C8BDE2A162F999A8C3A99BE998B9DB7A14C101CE78F08857ADD9A96059DC22592B90"
00823     "488D77D6F0A6D459CA65FC947D742714A42277A79745C73CB157BB825E65872D9BEF5066C4"
00824     "44229ED1A097AF6B0308DF2793A2284F911C764BD1158742EF4109CD8AE6111DCF8A40BE03"
00825     "409FA6C430829D0C9BA28B466916C67311F8BAEA2D8CD2DF9C95094E4FC6D43F4491AC5FFB"
00826     "F05BA6B7DB9D7FE4A6DFC386D136B1710DA5F30E67C8DE97A981895476B8E4E0DB5E1AE630"
00827     "C8034F949AA4095DC81EAA0F645B0B440138AA7C5C7C5FAF3B57E1504A5FE477CD61A355D9"
00828     "AA66C39FF89E2DDCBC1862E19102205A9F343FAEA71D4C67192D4D5A1AB08BFADAF6753244"
00829     "BE5B14B1F911700B4A947D53645DCE170401135D0D7E968F03A9B19D2ABBDC94BEE623FB64"
00830     "C1DFC4AD9040197C319A79391016D6D44B8EBAAD5B2F1AAEA58AEFA21B5B3D9ECB66B3B706"
00831     "26AB31CEE5CA759499073B4FDD96457392E58C0EC871B95C7B77F74B81C4CF5B1B243BD35F"
00832     "4D540C1CE878E80AAFC635D363B9D72DFCDAFF62FA8339BAFF4DE1EB3E146DB93E7E444403"
00833     "E0D4290AF501C841B66302A345F5541B1C797D630B5B7E788A793192CE1F362B22B91A8371"
00834     "84106D852391A241A903E3BECD192C4FE52586BA4044C4FBDB20B711D2624B6CC13E0C5EF6"
00835     "2F2B2E00CA516991C7572732DB94DB809FA0535D2E8376685506186170E3702903A6D08996"
00836     "BDE20F74BEAD391245680A029FB439FFB96EB829F0BFE58A6446B00B4E1894BE1B5017F706"
00837     "156072E2DCFAB25A3BED2DB310DA1A7262ACF24A65F41FADAE3F81A32077BB9CB64AADCC2F"
00838     "92B95EE56F20CCBA24DEC9F5F75B36EF1A101D0F3FF6D730DC199869F8F26DEFC6F51451BF"
00839     "EA2C0EE77CB17422E2790A8B29573405F112EDD704E3F46C7CD8ACF0E495C6BA9ED448B0B8"
00840     "9A9997850F09A813FB23788681D3CC04996E931CB00A879E2FFA1BA6A63F9D3BBFC2040524"
00841     "4B0F838A040F125AC2B4999A197A2D53DEF8CB167FB7E37DA3862CE3ECB57B081DCFD98894"
00842     "E45C6019B9229315108E37DCFC57C9F80201486AEC",  // PRN 22
00843     "4368224792A5CD4ACD64D6956A334DB4614EE5EF8F98EDE0B4A5CD3D06819A4858EE0037DE"
00844     "64ED93BCC7FDB47E90409638B3789BBC63134308B972765565651902A9CAF4F310E93BB08E"
00845     "F1E4B325D414DB5DD7CBF660A880642A27378A7427F2627BAB3444149181C54E9D3FB7967D"
00846     "E33BED89AAC255A47482835D17E329ACF3F08F23AA1DDEB730D9602243A6894989EE2D900F"
00847     "F6AC0579864C2361EA56CBCE2F39A75A12100670AF3FF1A36E98D1931D17CA68D8D031CC52"
00848     "7EF7FFFB35AD6E046F4AF1B2B3F66210D978F8607039BBA99AFBC6A499A5A948C5C59055A1"
00849     "40293CA97306BFFCEE37CE77C114478D0F73D1C8C9954780830D56878EB786967DDBC71951"
00850     "ECF815CC4266D66F84195CA1AE0375E90017FA5C66DB8329D668BE6C0BFD85742088B7A325"
00851     "6A3277AE2A1F4E72BBB82BB2A59B08FC59FE5F24EAD5DF43D95917A144B4DC6587C893314D"
00852     "5C2C427EA92D02E6922FB0B60ACA646092E0D4A127799FEE4B76587F26AC1EB89BE0BA46F2"
00853     "A212ACF50D93D3345AFE72AEDA1A0F855C45AFE14161E22DA3E223CAD6B47E591896F4283F"
00854     "10C4073AB524CF5D0ED88F979458BB928F1C179E2C5914B4CC52CC000383D154830A54332D"
00855     "4C0E652F98B91BC086370B04CCB1AACA56C9A33B9F0A65C33467BF45C74905B67607B83FED"
00856     "A063857AC4E326B4293483C8841572174CC8A384014945F5C3072C5575A5BE322E3F2FE3A7"
00857     "E30F3F21118FDDAF23645A40E0A359A3572153F5C3ADBE1BBC4EBA5E063E9F73510C5D1D43"
00858     "26D3D7A20384724279C3B4795C68BA332EA706E8127672E41DE4BF1E8B5EA60B39CAE2C3B7"
00859     "B21A225115BAE051FB0401A69A7FC56AE4B7E31CE2CD575D82C8279E8B8499EEDFC134503E"
00860     "6798056324B17B8AB045FFDC2A35B190B39FF212C17C36537A1F99F2EC3A4CE693DCEC7DA8"
00861     "0861FC6F3D064B82CE5581F9781289A085A300CCEFB6E80A86D961408250846A374ED00F53"
00862     "79F446FE6E551F608AC3D71346B738C75DCDF63E19EA79AFCADD5E1E07E2F7C0AC709F6428"
00863     "FEA1DCDD835E3F56B026CD8DC9D4629AFDA9015295C2CE55DFD9C83223821682B5AF876B1C"
00864     "E156343A0F4C1655448FEDBC992602BED5DA4B299B86EE2F86D02C757CB5696B32D60357B2"
00865     "1F9DAAA06EB3BE2E9117D2C99DFBB5328B3636F41A06A020E3CB47851891AC3825369C037B"
00866     "0C0D8412204FC3D5FA9585F3218F01295375A09463AEC65F591A594551E19DD00D5738086A"
00867     "31712AE9FA7BABD4E9C10E9F4DA82C12EB2F3319E852B96E61EAD2F85BB6765DAF874FC8AF"
00868     "42746F14B8FA489DFAA9540B4D0470CB31F99B10C335F672C2E06E8A2378B9DD28E8F56BC7"
00869     "A4C0173274O4C79F1E8E67BCAEB5C4C793C176F0A3C1833007FC44103DCC6E273C9FD1909A"
00870     "EF1656E2B5756B7F4D69F4F2EA576228DD794777BEB2F34296691A1FC38AE44FE60A44AD89"
00871     "4CE92DB77FC0246C779C13E8AA16648843BA68CFC5E4E02FB564B0ADF3163C5E167C275A8C"
00872     "15E58659E2F70A3AF792565422D3327CCE25C9D488EF3C32B195066BD27DBD44EF1A20AC6A"
00873     "A22250BC0592AEC734477E1112DC3913F6B7CE323FD1F7B3F8299B8C9BBAC23487F735070A"
00874     "FD5C3627506D4C96B0A1427F7EFC07EFCC62DDC5455A77B57D3FD0B8CAD30A346F8C34F42D"
00875     "87827FAFADD54264A323907A89334C0E7F98AFC9060D7767BC9D8D261BC0BF64FCA389BB19"
00876     "2A9B4B00CB566D254CB0226FA17F3BF21E2926CA52CE5DDC401E74A92C7989552CA66675A9"
00877     "BB9E9C198BCFFC488BE1DD5E47DEDD94015BE09B44",  // PRN 23
00878     "30F77B7E7E40E42A52452A26283CF3A29B02DE3398DE5179773F2CCA4978059495B9DB1306"
00879     "9B00107462C0034064A024DF3AD33C063F9608783E79E38F706B92AB117029EBD6156CB5D2"
00880     "78149875B95AFBE877A21AC3128BD6D0B11D31DE7A2BF7CCB68911309BBD2215F592D56AF0"
00881     "F700D962E49E4C6D089548C7587B0598374C3D7BE80F7F20D6855AA28A27AF668A8AD3552C"
00882     "FDE3CE0B35DCEFD5240C8FAAB89D013D0DDF5B66E60328AE32BF4EDB8021E43762E96E5999"
00883     "CF23BF2457B1CD75D1457D4926E7C2A324BCFF70A7CF59047DF3F08EAF2922CAE6A7BE1BBF"
00884     "A3BDDFB864AC965B7FF093CFE74B53DB332B411962DA01D4E6C62CE6731E893574A097328C"
00885     "A7606ECF373CC987F98C31AD74EB4A863084266869D28C07F9C5B5F10385EE44E463324A81"
00886     "D523DFCFBABEF20F3F71FDA33B7EC1B2A35A98F01D13693CECFF23E00CBDBA2A2D09C8A86A"
00887     "819E82D781AE88B9F207889EDA1EBC54CADE6A69A564ABF48F950FFBE7A86243C544A78362"
```

```
00888     "D9BFB6CCDF20CEDE386B9E3F8D7EE0E4A58596F095AC1B99E4F787472A734252D6C00B177F"
00889     "D2B99A85D49B9C0F32D728F7D9BA4DF96D1BF6C365F5F4EED7D38727B339A223263B58560E"
00890     "CA6778069DFCE48703AAD1BFAD59CE298C6BB464552408946265B1E19C1FFEE000771814FB"
00891     "5E4688140E1333B270F05EC9345E8BDFA7B412F295AE3DD4DBF33D6D2D62582887D4CBB9EB"
00892     "55F5FE3C1BC32DC689548B66893356704968C143AD3CC2E48BDC154620ECADF336E218FF1B"
00893     "22521D183EB2D13B15A15D39A42A25B46F1AEC07A14CB805AC382DAC8CF96829B8F1D241CF"
00894     "5CDE5D849DEB242295D7870432EBB642A45A6C6F1C2786B88C883B9B47343E822978155D07"
00895     "9AD9F6C8CD4A21479B32B2112EB0121C3C74E7C23284812F74A2B413B9A18D6372AA3E09BB"
00896     "04A262E7FCFB6FABBB074C7C603C89FF607F83DA32DA50F2A19C38BA0965BEDC7A96196689"
00897     "5E17A20A4D8028FAF7B6947A1964601DA45643C5FCECEBDEAAAA83EA5420F2984CB606E9F3"
00898     "191EF323D644635EDD801130893D84DFB19E661E04188C6CE596DD38C11D1CE5DE1EDE0D15"
00899     "D8FC427842085D300E6801CDE02768E44BAFBB302684FD945C9AC722561B0EDA92E8D845EE"
00900     "B99F6FB1889328C74D6D9CC6DAB59D55463A613AF33226660C0C034B896075D9EC2F51165A"
00901     "754A7D2393E15955BE327587943ECC1F3BB97180E751941728B94948B6648D2C606D66B9BA"
00902     "1898FA28EFA38A413B066D715F7DC5F0C755B6A3126AFCB606EC6B12E332FD71E386ACDDD1"
00903     "DEEF1533D6386AE5994BFD857380874B14ED836D9DED47DE4760ABB05E80DC622C8D7882CD"
00904     "74B2720884DA747BAC3ED9A6EFA2287D5F7D6AC7821E79C79EAFE7A68D2543D7CDDEDB63A1"
00905     "3FBA4B894A6D0B3E14C44876364C9FCF93DC066CE953E89851A49B6EDB38661C9316C82ED9"
00906     "02EEB894321C84331F6646A5C1AFEF832F38AFEAB89D181D3DD1338DAA236BE0E87A6381D0"
00907     "0845CCFCF4DBEC3D41F8AE1E6F39167A499945DAFE77601147EB29804A8B98E7EEB31DE20C"
00908     "84E9148BCCB15CD02D1E3B519EE0A2E4F70F57A69281E651B189C08D575D52E997E3A435B4"
00909     "CC86F938BC79EF4961CD03DF8B92A8C507DFB2DE4D5EB0ECAC2F3D712A46CD3EE01B8D4DCB"
00910     "845D23E53A2E01DF3FFA100078F7B2467A45BBB4A18D14E0CFCD089433959F07F6F60F651D"
00911     "D7FD9828069D340D048FC079E4F8B92824D54322611A3698FAF181008DB5E3E327EDC431A8"
00912     "F6E481484BD5A429D0F5DC55AEA8A18F719A7865B0",  // PRN 24
00913     "A7D629A25F82015CD70688CC461244E4AABDCBF2CFFC2D73A3247F4264101FEFBD5759FFB8"
00914     "3E8ABE9EF5B6662DA3CF7196385C05D0E5BD50C71D3CED424AB805AD632F2C1930868ABEDF"
00915     "493BB4E3A9DE538D8E35146320EB9F5B71C40E6F32E9243E6A25F503B68EB484ECB73C4369"
00916     "30F30E2776D17A2E5F9CAF157F368E0FD9C26A9E9AE30A30E15DDD8A56482C85B46F973D00"
00917     "0292CFCF0021577E5DCE8220353E23DD8507D0F561BBCD96F6FAA451E9C595E00CE0F99C8C"
00918     "40359B103D621CE0C837A3D1598D814BE6E22B7A509DBF1ABD2EA60955364791A591041464"
00919     "6F9928B5676ED590F41BD128A55549170211A0630BF5C5BEFEA204896C7FBB40F0C72E40DF"
00920     "19C95F3A0ACD641834E5BCBD26E91315DA34708E2025E0B7D38F9513945DE44AEE57077C8C"
00921     "C32C707D9A39BF098E4928420B3FDE8C74F39F81790D827A628B8F2776251F51095C1193B2"
00922     "FFEE8C93F49DF6D4E3C2E1BDA52F7356394BD5A675543493E04D602F332761AA1D57DF3EF8"
00923     "213E4E9B8D4BF6C8BE8BCCF661507C394CC9D5E9DD4D81DEAEB6D638FB9225247818710E47"
00924     "24111F522610075AE757AB5ECF6FF32CF087DEC9A04A7DAD8C613B22F2AFD31466574AA87C"
00925     "F0B8545B38C53A8F4597D5F1BB6837AF26114D72C609B624252722DD867572D449ED7C170F"
00926     "99D4ECB995B76C8DBD14F50F3DEA077FBE4F838A03AF1DA6757C70C5F3394092D7EADB1E0A"
00927     "859A8487F70D9F7EE523AFCA08DAA5F8E9260D63330773FB7EBE01373C3719129E483DE3D1"
00928     "32697EC4C108654256315A38D7060C6F52B4A444EF33E87946C2537A3FB3B5FACBB0F3292C"
00929     "7F2498F0A11509FD9678CC7C678D4623DC07D8F218E4B65F2EB844D24D696A31483E1EDD10"
00930     "63AED60D7A110F7992E3E46F8C7D5CB1AF02D598440D79D649AA415E5F3ED033A8FDCADF8D"
00931     "BAC48F9905BACB37E31ABB0174CC08A9584E25B94423BF395E097E2D0B7ED76B55C6231F18"
00932     "0D1A7BD969F6B874B9255138A1C3A2847EBA159C7EA281433A39D2F88D724B6350E846B365"
00933     "BE838F25B904E7B02B12ABE543961D33D991D0DA93310CDDA1EF551D7B6A02765DCF0D870C"
00934     "2BBF578BF8457CA1B599D0FE445B1E9C39E9C8756EDAAC6A10F62C84F60A27268FC5B436A3"
00935     "23AC5BF4A9ACFC55510558C5A354D96A5D74F69687385C123C497543EC14C40117ABE46C22"
00936     "FE54DC4D2D6F4823A7F3DBFD3A1FEED428B90945B8F6B2C9A162F9C0287C758603E2C7845A"
00937     "11DDFC37B815A62A1A665C3FC003DC9F064CD5D950DEE02F48505CBE1A1E515E6516E926FF"
00938     "6DC0123E1E4F94587F1F1568DFBACC3728BDC1D153196CD91D9F6DA84ED6B70D0DBA7E7DB4"
00939     "A71B224E5122328D1DCDB9FAA98E88E9823A49E7560DF822D67C908E9AFF7D429E84350163"
00940     "E49AFA4015337532B6E928BE2001CA202C5D975A3E54808AF8E110575AD5FCA28185B2A7D8"
00941     "A3019B97AFBA933BE2F9CD5F279266EBEF52E0B92114AACC30237B712FE53DB1B8596CA72D"
00942     "C82BEA99DF8FE845D390B8DE1C00F24D0D9C619651FB16A280102CA8BD7FC3F7868E4649C6"
00943     "36064F4D0EB06D06595A0023E6C5D32654EAB9C8EB011186A32C3E5DB9BA6489895FD50A26"
00944     "CDF1BC27D599F7A8BD1922B30187E0DDC54AD369CC61E958661C4FA76054C2C6658AECAD7D"
00945     "0AE23A9F36DB1B0E1141ED6A1084A8EBF8975325A1DE4A679BBD0E022BC35A07B40C52FA1B"
00946     "0BCDFE17BB6EAD099088963D7937CFA5B51ACCF722FE4F9DF99DADA8CB05C17B6A6AE884F1"
00947     "C7DA09049027D6EA570BAA064736219CE1DBEA7DB0",  // PRN 25
00948     "9BFAC777848D427BC9D39876B843378E100DDB87ED269AA91097FA433CA7EEC7D6210E7F10"
00949     "0C449876F83439E3E494941D8CFB3E11ED7AF04EDFFAD2729FF75A8FB14A9BCF386ED8D95C"
00950     "046A326A9A92D3F316C758F6939F14BC88789B17F3DFB5AA691A872CA44E908AB966D5F672"
00951     "FB90196D71136ED26591404202AF5858D44C8E0B0ADE3441F554040E458F5AA64AC3D5B323"
00952     "99E453FA1565F4E71FC9EE5BAAC97552AD75D0154FDE48AD914F6D59800A7FEF91BFB6465A"
00953     "296E217E554631FCAC48FB7FDE1244F00697C2155A3161169225109AF5AE218D51D70BFD45"
00954     "2433BED94FAFA5920B35353013BE2CD0C32B8AACDFAC6C6B4CCCC2C222AE845FD76630E963"
00955     "CA22F5214033FEDEE3296112F9C127865137CE6182BBF09D1489A1DFFA82D9B391A5AFAE6C"
00956     "80414698703EF03BA42893786FB10C9DBD21E7292E197A2959D605EC466CD202BB4E7BB1E9"
00957     "D5B9B3F07F55FD8637314E3D68C763CBE79982FCCC0147AE6D4C96F7A1C251BC35D8EE3A4F"
00958     "6FDD9CEBF771878B80E3AE481619D80B7BB7EFCFA9C20F11A0FE18505299EF1BB195EC7A93"
00959     "263420D183BD89335411CDD66B4809BEB30AC649CF76552A6F08E9D77C2E9C5AE05FE46970"
00960     "5E2E6F44485395C054F401DC6CB9F581E52978BD93CD927D47472DC52D81960F98D60A061C"
00961     "5ACBF9F5E4434B31BC3C55CCD71CC2F2DA6385A3D040DE8B8D5F5CAABB6FC301FE3D5DEB95"
00962     "A00F01772505201B33464F2BB1D45A2F55EC907E86803A94F9340960A5E99C727517C520D3"
00963     "38A41320E7E8BAAD6C4470300DB904B1848AC84F67F67C6BDD6AB71E2531C5C662AA9DCC2E"
00964     "84634A5ACA48C5A5CD1CDA454C6689E2484274EB1A2D1CE2D94876F4978987331D16AFBA20"
00965     "5E1D81CF287E5E59A6DE000FFFC99E50B0B124E718CAC8FB51A708B42C0FE669894383C01D"
00966     "020E45B301DB2FE85B177FCDD1A688264A0327614A4483BFB689DAA49E8F59BD2D3943BA06"
00967     "F39119D46E5DCD0DEF78BA40798384BEB03292F231A62F7B6E456F3542425599B2BCF96978"
00968     "626543266E8AC0426B3C4346AEAA1E2FE20B54F70273A5DEB9BDF92B8B4DE940C48963DDC5"
00969     "69FEA89B4A6940B7644E33D105C5BDC61C3EB7D823D4CF3FA0AC7196D6004CF6654BB4E460"
00970     "2974FBFCF6E74FB58A83540254EAAA4ACFF41C5E1481B2C38E7084B944684B21B7C661F92A"
00971     "28508F449D8ADE8881D40AF4A161A08C5846C78FC5C562CBD36ACB83284223B2891F739478"
00972     "04B6AD40BB99A4E7D9ED1912448C99E5B572760E53DFC434D06DCCA06DDAD3BA09DE063378"
00973     "A98943EFA0CD3E75B1B860E8D1350CB920BD6FBE30D437B96290F49E37BEBCD2FF9F8A4BAC"
00974     "7CF8F4571DAA9E788A3A411AE231D369D35055EA48D03AF54D968EB71F7BD14E51A1E617F8"
```

```
00975     "1B492FBC3AF3761C601CF108DEA9E198BD1B576F94DB9A15975CA55D356C209E5CE1D8329A"
00976     "5D353223AE3C5C9BC55468AD4C2756BB54B6FC66FCC3B0BBA3962EFB4ABA17B1B08B01C296"
00977     "A578D5AD63F605BAB75C90924AD456D53CB72436F3A2086EE61D7C3479E0381A52715292DC"
00978     "B8CD2833720572A40CFF97928D4EB54265F3110714C360A0479AE7D51126C5CCAC09372AE7"
00979     "139C20C46AE13E4C41E89AE12097C6A198A979DCE9FCBEC5D1AF9D9189059E5E317B77B6E9"
00980     "8C29F38E29A35FCC3BEBE3ED0267C5503BAB0FDC46C7D89766475F2676C80CBA71110F815A"
00981     "84CA7EF91ACBA63AF7933056DDA345D527DCB5549C9CCBD5B201BF9D0C82683A8865DBB4DD"
00982     "D057002025D088B7121C118B3A6C901BFF83DC1114",  // PRN 26
00983     "18A25B1D741562F456B84CF500F138FBB34AE1D18E9BD6FBAD05D0AFA14BD1FFD10334E6AF"
00984     "3ACC7AC2698BA10465ED5EC26B13265E5ECD1C79359095F36B1B5333EF87E78FC44571ECC0"
00985     "0F294243FE64C4BD7833FED71F3B6471ABDDBBBF4C374519D24DEB11A8D2EBDEFB8D2CB3BF"
00986     "517AD92CA598040D0F5F8E5607CD879C75E15755087EFF796EF432073AB647C9DCED072FCC"
00987     "9DC74D78326EEC1BDC9CD3272180E38DF556715CD1D37DEB8B6BA06B81F01C26A3E97B9475"
00988     "B688D32BB642EBBAFC4406B7625944603FC085B38E5E4C9ACF0D0F92A65EC9BAEC8AFD9A1C"
00989     "7805854B19EF516BCF65125603F131084CD13BA6E72201B13C1897D0ACCB8C4266884F3439"
00990     "F6833D22C08A68CA0EB7B636D866182E6F42B35EE9E255F3FD30014F4EF3484EF31FB1D17A"
00991     "6FC0913DCC1443439494BDA221336D05CEF0FF168A9C54830066A6E986672EB92B2CA01C7E"
00992     "CABD5B5B40049B2161F8F52445FC27379B0706866AF20800C521BD0DCAD90E8F4367929A9D"
00993     "21F1921049FBFFCB2BFFB69B27A2A4FC9E48256CE24302A57B69B04D6E51469D5D7C7E41DB"
00994     "6113693322E1DAD218FCBBFD47892E8BFDC28FE1461AFC769286C6F75165CF945E5CD3C932"
00995     "2955261922B6C00CA2C006070EF26141BD8B44798110A9F20E86C52CE88D6FAE122FD7208C"
00996     "9216011D7D91750A800576F575B8EDDA2545B3D60310D61F7351CA1C82C90E18641FCD747F"
00997     "E901351524B9D0B92CCBE86067EE5F126FF4B2C582B280CC1549CB3A18D5D9FD25A09A67DB"
00998     "83A8F3F7DABFAA3C09CDF00C6F16D722EBCE259723B97017885B70CEBBC1C164E1A5361738"
00999     "B1F3FA426D077AA97AFC56F4F147D3EC04E52AF0CC518AB4CE8B3ED66A1EDEA49E904B81DF"
01000     "00EEABBB5EA8437C18C0282EA1AEE445331C5B0414C2E3CEE637933753B30FBD2160E93C3E"
01001     "7956BC032FF839B81179DC5C41CC81E6B8A74EDADE4F6EA0FC2077D3AFEE342D75DDEC1539"
01002     "F5C00F640A56EBA718A77A88E2E8070AB9663037AD66D9B6314C268806C2856E233D37BA73"
01003     "E61B8CE393F9DE86DF5C76610766DAF68A8DADD37C3146F59115E8AC18330977D7BBF677C7"
01004     "55B79C902A50DC457FEC314323927BFD45DD4A5AB7AF351228986F3B03E51581C8A8D5E4F8"
01005     "C24ED9E14FDB5D0D6BD2A6F563D66A31247D4EA17CF6C158BE7FDA81DBA14B9A1549D50926"
01006     "58BEE5542BCFD42C8DB540447AA13F208223DE36AEE06BCC2F280333D81964ED05A275EEF5"
01007     "6C21F0BE0EAD9D6A8F2C70660C0CB04AF6D84E0D18D3EF56504CB18EA259BDA23AC1D8CAB4"
01008     "2BC340448283D9CEF770872F3B94FE85A23CD9A864F76FA82EB637086D12244DBBB9E7B295"
01009     "1D3B0B6C345E513883F261D3EE312A31017BA33A08586F58084A8087144CDF634517F42AED"
01010     "F185D53366FF728C0E7C0748A776F3932CA874D3C2C48B30D01A5F579DDB1A038AE1E20D98"
01011     "6515309A13B369F644A856C995C2F6942B7E6BB624A0DF34FBF31CD9F96C486E530A2F3FF8"
01012     "A43D7DDD578D72B1E93338EAACC244650F8BB37AA9CDF721B1EDC334755B63CD34AAA83272"
01013     "5A883333A4AF4ECE71416146A4F05232968680AD41C89872E14AE1CE2FA65B7A3BC099FE26"
01014     "5173B0B7159662259FA64FB70902BA09988A30697F110B892D7D6C763EF80D6E904A2CE47B"
01015     "ECED5E98AB2A3BB70A73D6AF7175BABF8CF8F38F2AADE3BAFBBEADF208415A2C2FD4A1ACC3"
01016     "140E185A2534AFFE6C01EB30FF0137503349EEDC1F9BB263C18F427E2B119BFC692D5FF444"
01017     "21208E8AF146FDE964C2DEC3E1C9D5699ED01A7D78",  // PRN 27
01018     "69A39F2889ADC98D94B5C678A3D3D8736BC17256367EEE3FE8EF5AC6453A0691D894F1F320"
01019     "2CE84EA8BA5A341D1733C6BD0150CE9C0019168E67768B33B64DD4C9F0E2BCCCF2FF7E370C"
01020     "2BDE4F403C270F36937119950121A840913CC1D663F32F01B665840EC9E09E5A9C345EE001"
01021     "F84B2C1154359BC066D29E0405941476C5B7B11CFF124F5324D4D090E2BC49C3E8D83122DA"
01022     "9268FF58B523EAAAE4493217609FDD5A5C510038CF383B65FFC7671119556702F630404C0B"
01023     "C686975E853C86AB392D25C853964693AA7F4A63BEC12BF51762DB46295D2062FFF24000E4"
01024     "F9FDD21972D747ED2F37DBBA2D6DECE1009222EE06B846FB7D49E8014DACFAECBEF2560CCF"
01025     "5E06EAD04B113D1F8E341F200DD8CDB3F36AB99B566B16A84F8C040ED8827F0E9F0D4FA340"
01026     "B0ECCC98F5CA4B05A990F3B410CAEBAFB409E985A722E4C442F59AF3ECEB1AF0B148A3E8B9"
01027     "58D9F6C0E45BA4EC8553332DF170872E60967E106BD27AAE4A6C15D391C1B4DF7D77357B4D"
01028     "4E8B29FC874947A75CDB1A1ACE512C5CD08405F2D9F21D50197DC151B5E5A735F759628BCB"
01029     "5A70B5CC9855A39188FBBA8E6BA0F5B6538ACCAF394A809C68F810441BB81CE71F7DBB2A11"
01030     "D3905225DC9B1F0F0D8B7E5268020039E03FD97D42204E503EF98C26B5D9DCCF5424573775"
01031     "E536C4AA26BF4B0351530857B8F640495377E9A08AE82F227BA487FA1D945EC233EB1B42C1"
01032     "8FEC4BEC651D4CE1313C9574976F092A5E52D1B37B5D4B33800F5C3803BC6ED2759AE3F688"
01033     "8E7120D3AC2F24729A08B2B4C8D44EF807987C156DD3ED871B0A6217E9783A78BFE56C819F"
01034     "2C8FFC2D2D83945FE846833E5ADE3F840B53EE67E5D13BF0B7031B6C1D004D274F43C1982C"
01035     "43FB58011389D2C7E9111B8ADDD89C37756F1E4C4A931D828ADC3597429AF3A00C0C9A71A4"
01036     "4A62F922102CABA479BDDF25A088BEFDF630E2AED4351A90B9567FB6DB5FECD6F168966DB2"
01037     "EB6161DA84479DBED9090026C3CACC19B91EF5680A40073FEAFCAE161E3A2F0DAC9C0A5A7A"
01038     "A31D93B425D5B5A7BF6D38FE797FB53793EED791C82DDFBB2D224455349F26F147FE0D5974"
01039     "7AB091585DAE61A32B04DCEF19D907C1159FF69443843248D75E772B558E7AA0FFAE2A9D2B"
01040     "6B1E9F34B79DABBC98B2FAB34F76FEEC635E6E756F7724106D08F296019FDEC94FA7B7CBFD"
01041     "780F2C0B75B1D88EA0D5885237F4D09F03B2E8E1A98879656AA33AB23583DAFF96D8FCEF54"
01042     "8302F323A8BA959710A78A803185D88A9C971D9491613E5BA2044D0B640FECF219CEFE3774"
01043     "45E5256FD687A15C9F66475C0270C9C1A7189507E881DC6FD5B31FFF99493425586E24C7827"
01044     "F260CDAE356F874CC7C913ABCB40E892E495CD4388AFB84AADB45069F37ABF7DF966CBB8EC"
01045     "86C0CA5A86E79350DA5D6BC2B23EEB51B9C9C352FD9D3FFAE368891D8FD4E2861B8002BB16"
01046     "DA0C4A96E6A9B8D8FF553CA1E6D85B16F996FAB677FE58608BF58AC1AFA9482CA957B03924"
01047     "945B4260A4147625EE657C7763B4C9F4C00466394DB4F222DDC16B72C579FB80B3861D6B43"
01048     "64007FE856DD8D498F3732AF3A7041A2D71498ACB3793C250CEA60DE84FCD0C227A08F903A"
01049     "0B5ADC936FBDB7BC6AB8F7AF002266C8740AC3D75B0628EF6A05C0481CB283E3645F7D8A0E"
01050     "D7415EECD52E4BF70F00ADB11BAEB947F03140446AF31C2BD41F1D34D75FEAE64B91C6007C"
01051     "ABC91A07DFE8AD753CE183C611178B8F63B822F03AFBD41AEF17F6A1D58F341D172D4682F3"
01052     "8FED397FB01B396352C37397991D2E1473784900B4",  // PRN 28
01053     "39B27D6129F6C84173251B68B742EACD5A1BBD49DA6A4DE039831B66B054A9BB13299EE828"
01054     "282F1779AAB7FDFD82F4CC3296E9777A03C64AC4F08DC8882A9454B301264284F7226A4AAB"
01055     "ABBDC8336CA2BD1EC6517A9F54A776446C5A12FEAE387629FE5A66188C671C80A6C83B0832"
01056     "2C2068ABBBC03261AAB8280F4E36507788CDAC4D3F0817327FC0D19B0BCA35D89819A9DC2D"
01057     "5C0CED08A87AF8DDD99304505EA08855CDB514357F8A9A7F3F4D062705E7B26632CA617F64"
01058     "0F4D99417CE5C90D432D6B8955DE269460B9DB3C08432FB7BA08F5A658DD79DA8E9AE5B075"
01059     "1152847EA6ADA78D0BB4A8BD9F6E221990CEAF771B5AF603F463D2369DF9BF556A1CE60D1E"
01060     "A6172BD05EDCD166255810C1FEFE029EF21147DFE8EDD5B333F6C7274997E1BC61F253AA37"
01061     "5B9CC6B5B64BE8DE11CA00C9530DDB1F218870611D4037BBB88F257614EB7F2A6FF395376C"
```

```
01062     "5DAEE4E83C5D11BD6600DEDA47D1CAD95B11E5BCE680D2F18CF92353A11840DB88C38B363B"
01063     "E8813E8004D2BC1DBF95E86C64E060F735C404E8945EAA3DB31A27C63AD818D96EDC6B0111"
01064     "CEE3B3E9F0E1B7DAAB22364FC1B36D25ACF77630DBD1E3CDB194FF8B8D1BE8ADABCF4D0760"
01065     "BF5CD406E982E0F743AFA2C4B333152A74177DD79E64566BF34A10515AC0E050CA135D7357"
01066     "E1F94F1653E9EFB0FBA0BEA1C2B09D6378E4E33EB98B5DB2A2DDA5D9F29CC68766AACBA09D"
01067     "E25B338BEE82700203628ACBDD4B828B34856910D0F2230260FF41EC371D08BE96EE2E821A"
01068     "1F3F328B631B53BECA81BBC74A92F608F2B040548BB5E805EAAB46815793F1D01D121D2E34"
01069     "7ABF1FE62B4E00E31C8DAA01202340219430E88938D859D62B709D43934265B9DA9DBDA015"
01070     "37F54B439111126613FFF46ED0A39B2B19C22B8776618B065E15A515B752E1846F51F8469C"
01071     "E32FA076F47A5C49541A19D54CD63512DDE1126D7850B4300610E066AD53E094F26DDB4098"
01072     "0B47D632DA9B4524EE5473B521A8F79A4B84CAE714B65E30824068305858CE9F39BB2EB068"
01073     "21C3C07EF2E030DA4BB3DC2D7A29A69DED7EF7B03825C663F5EF44B9A1A174ACFBB164FFFF"
01074     "F0419E13A8DDD108FCA9B2A8CF1A5A2E0471046FCFE32555E72D857ADFBC123B190B48CFC9"
01075     "1CE8F0BF86CE2B439DCD3AF6F2CB26EEBF393FFB34290D2B7EC33FEE285547BDCC922312D3"
01076     "6D9D25AE5DCB0A03EB0B603A6F75A5EE878266221C9B606B185250231E4B5C8B9470E5BAAA"
01077     "9859A963A51B83ECC111DA4779F012AC23AD2E2DF2965A39E52BD684A0E8D2406A173465B5"
01078     "7BA6DC095298DD716BFE30E42CFDB0F2202D56E97772E3D64F1CA9B628B6EF2217A837165A"
01079     "1BA1A8612638C8387EC3854AC557C326CE88DA10F9DA027B204CFE634A695F2B59BC57F987"
01080     "C613131DF2F0F6A5F6081C96098DC03F82B3691AE9E4F14C32324CE495FC94EFCD1CBA8884"
01081     "2223DDA27999E0B1762C581162FD1839352E8FB7E4FC143E8D3C774BE9BFD7BA4C3091A04D"
01082     "08EFCAA98B5D3AEBD1FBC65BF1CD7BEE784B58A877D8AE9DA9B3D459521A53010D42242DA0"
01083     "181F661F7CFECC2E74BF3C8E959D84F91904F4F8DE602B214A2E3C42B93BF904058FDBC375"
01084     "28D467E323039F101209D60C56BCEA60BF9C3714A05437A978F3601FA625B31562B5C9B5DE"
01085     "2C6B84F560FE35296E2715D905BCA71BAF6B577E7A03CA4A5DC816E79E21C9B688B0857887"
01086     "A25AB624E2233FE8C9A80C6FFE9AE77108F44A72AAE91885BA9541939A34519C457C41964D"
01087     "4FBD492E11B2890BE2BE0E069D274621FABEE60934",  // PRN 29
01088     "4545989EE35583C07E91C0FF1F5FD605F39D51A0485105A71ECAF9C7EC0748F7FE5FB5C4E9"
01089     "0F1A59B1E31F74912BA831BCB9C95A315008F4C7C72D8AFF51DD09D9E4738E72C325B128E0"
01090     "3F7D90915066AFC3FDDE335EBC4EB34895FB9807731EE6E123A28E815E5927BC9FEB02AA96"
01091     "0308978D8DB5E131A22CD601D88069D145D4366FAF97FD7B71F4B39AB73A97C715BEE20A17"
01092     "5931D7108AC1FCEF63CBE82D9CFA3569883A448220CD92C433ADD11A52F72303BB4A5D1DCB"
01093     "DCDEF605913D8D424F513EB8920EB864641B5833A39839822AB8ED011762756D6E16540D54"
01094     "4B5CC8D9F8B671F19DE2BF7ED74CDF8EC9592CF1FF61095D0749A6404BE49AEB7434C75379"
01095     "DC02A87D213FDD6E0E6DA751657D7D221BC5EFFC8211F6C86B0F77096A81B1F60F51C23A5D"
01096     "525C3411DFF07CD5FCD54A60FE7AEA7E4C182885B2EF08BDE99995121FCDD5DDB0335DAC07"
01097     "DE1AEDBC9E988D7D90CD81D30E6C8E598BE6743D2853D975F236E6AED206846FE8F6AC9FF7"
01098     "2C585606FDF6C25F5CBEC0C47567F389D5F317DA01F4A801FA2E2CA60344EAC2CCDEF92746"
01099     "08D8EA0A0D6323A1510FBE7BDF1EB8C0F580DBD3BAD17B268BB6E232B109504CFA3DBA305C"
01100     "7F273F8FBB7D8880303492CBED0B8AE374ACDF582A945A1B1D2000E71FCA445CAB59EDA897"
01101     "7E5595734A508495F2A742E30A006D383FBB6F83BFF1D0DA35805B6CC6A54BD1A7C3BFE81D"
01102     "CBF47021D6813B0E28000AEAA0654D085A9D5A0F0E0BEF802AFCEACE79FAB3F85E09E968BB"
01103     "ED03B00AD50D079D9290F7E18904ACA46A2F0472E64CD16429547DA8A35CB1D9D532D42234"
01104     "83E7CDCDDD9E406B4920EFDD2E134B169A1DFC3F2AE54C3CE78A121F9F46F5D7BA6BFDD9CA"
01105     "25FEDAF0B2AD447902DAB3282AA9339C235FFD874C5E893E7E171CA838678C05936F9C9F79"
01106     "4300DFBE63AA842A3B9541B07EF486C4823EF1EC56ADDA3143B89052D624EBCA9DC5A3A68E"
01107     "19F7E50E4DD71F4B111B24459EA8FB211053AC3CEE16F0D2C75B68E6F264A4E2874DFA99D6"
01108     "C6B3330B634A65D1B3617AFD7C32981D73C8607996F107211A7F37C6D6FD1D72B56432D66A"
01109     "8CE99E30E16847E549E54E7A3EFDEDD39741A55E94B1B735238093F856D8690A813E0ABCDB"
01110     "630747EC7A9D4729A6E0A70F001F5867E9AF613E9A0494BBBAA59CD1D75C83265E49453071"
01111     "1F18313B0643890492BD13B49296827D7444876A98DBF99A6FDCFAF7883E02FD4ABFAD610F"
01112     "DD2BC8306377EDBEDA3EFE03041101A489051431B91BECD6E9DAA5633C8E99839DFE70F4C1"
01113     "56FC2B1D27ED307757E0335EAA699C3FBC43CF47E520CA1404054F38D13CFE0339D87FC144"
01114     "0504AFE91AF4088820F7A1D8128ABA53DAD634250CB0C467592C27D1DDF7C64E7E53085673"
01115     "75FF73C02951DAE125866C53F845852C5008A69DEECA708E5A621FFC81C3C932319D15E867"
01116     "AFB67B1B1E07B261ECB56A013C441A7DB94523F103E93D95A7A0FD95141A6342922D1A7130"
01117     "ED95F4B1FED18F3FC61315A1FF0CFA66382BF10DD284AA4B54B80FC2415CB8AF7F3E686AD0"
01118     "664C21EE9F20070F99A8A44A5D51D8CB4BA010659C3070EB3374DAF2168562D89FEFB10100"
01119     "670663E71D029C2B09107611C41B202B0815DA20712569889 18DCE69B35F46B8AA4F11DE63"
01120     "CE565AAA1172D7CAC8296B9334FAEA08D3BFFB34264D3487ACAE3C9781C1664E800732C387"
01121     "F9D65C152948FACFABD175C0FBA3B45F9892F8E98F2BAAC48E304EC9CB9D2A41487751B8E3"
01122     "7F3B58F66710EA2A2CE02354B85E56EF7AEBE52388",  // PRN 30
01123     "F2BC6298F0B96E3B736B5A6EFF22ED079C8AE2FF586E0F4D53ADFB17300D11CA7B8C2E79C0"
01124     "877970ADD0579DD90187CB8095278FEA5F4BD3D9123D300100AE63635463FB5019926B9B53"
01125     "BA7CC826A14EC4B5AA5E9B6C72685B4517A4564028FB6D5584994F8999F9AE7E59D4327FD9"
01126     "5C2A45E83A63E9D9F04B554C8906E0E2095448F75DDB68C8057F54630BAFEF06232F760928"
01127     "2B1EC5D0AECFA93E8A7EF3E6FFED7F6E4A2EB8B82D22C5B9AA7B8617DC13B69386742C09C2"
01128     "055AE5C2E71E73EE0DDF1D23E4928D7120B8C171A4F4421DDDEE6229656295430B936F6AE1"
01129     "8254A11612B9ECAED23869178CF6B1E6ED7430958039A7DA988C1DA03E0DF5E1D8EACB7C72"
01130     "6A9108E54723292801848 61ACFEB110ADAB5ED03ACF5EE7E806AB0C850AB7E2256F8178833"
01131     "94177AC0E0EF5B74E27CF6B315D0F134B282EE527E32A762C1AD06991C733A7F2E23776B01"
01132     "EAD68363A361F28F381F6190635A6BB38CA9461749E8E3117CE4F767007E2C38FC0EFFA4AA"
01133     "A693664F30A5704D369C9BEE225890021DF8450EF25B28CC123EB178D113B07B99ADC4AA47"
01134     "FE0078EF3D750B2E600B09353CCCC8BACFF198EDD2B3C551C212D005CF33763875B3753E14"
01135     "B14F96847142410DAD4046C9C8340946D922B6EB6CF2D2BF3ADA92A492D2C43397716A4BFE"
01136     "ACA01BF2051E029D5C664E4D9C3B7248A9142092DC25FD0DE44275AFFE90848463E7E68A7D"
01137     "C4ECDBD5325AEA355E111929BF700179AEBAE60E26AB2C6ADC3967AA55FC9A2BB1CAC49098"
01138     "C16CF85C569E2E4A146568C41C815C76629061A483AC30DD27ABAA1469C1AC6C527BBBA5BE"
01139     "8BB0541358FEAB34B1E3EDD7E37EAF307AC213055FCF44E65882CCC9A12AFE5F7FFFEF4262"
01140     "C19AC072C291ED636C3FB1D36D7B99F77E93A5DC2848B01595D8DD3C7DCB95075AC5216889"
01141     "7126055E8EA8D1028AF2718DCDA34496ED6FD1D86B49BA42F77A714D65D594965A748EBE83"
01142     "501D963F7B8D3EF55804B4FD5462D12571B77384E3D23C77ADDBE7F4BF667CECE0717C7156"
01143     "623AA6FAED9C1D2ED3C46403F750951AD8BD192FCB1A7FF95A590D085E7A733DB1CA0A8216"
01144     "31245100FF92A5B1F63F53A6D831EE041966571EBA5914D81F838FC3CC5C449BC39709A489"
01145     "CF31B7CB91ABDE6FC8BA6E4D41E048D5C59377EF4DE7BBFA2406C6F400232E6B502A10802B"
01146     "420A11316D93638A926FD0BE4A368A223FF767B11ED73E6DD1AFDA6508D4CA6EF3517CD336"
01147     "2CD1604CEC574DD35FABF6D5004208146C93BF179ECAE54D9794FD13C34053FEAE2E7286C9"
01148     "64BF2B2371D9BA8AE3F0114BC138EB3C304102846E65E7908B40358BEF9F290038FC645979"
```

```
01149     "951AD82D98106440CFE2261C5BDB6BED587C19EBABC54469D6ECA9D9DD59A92DB30868297C"
01150     "ECC90AA8228E28C4D1B074881B939C60D8A4530A193290C5E5399FADEBDFC42CEC5FC2738D"
01151     "B18FFE9FB20B7D443FA3F1FEEF91E65279460D27F6308B6B0F8B450452DA56E81D37851E8D"
01152     "589F9B0822ED525BAC86575FCBA54E67BEE4512A7F215820E1885F79A5406772A8241FF965"
01153     "33C644B1B5200FDDF4CD1F270E6EEC77324DEF0150183CBDE7D042C33DC0C27AC6F17D670D"
01154     "379AA778E6EF48D41DCB6C9F0F7C5E57BD0A7875D4B2D8E31AEBAA58AAF78947683E373AD5"
01155     "17DEF236B037F29C2A09AA4D596609B7AB761FC8D7F566A6BC8840CA626BD36B1D50AF7434"
01156     "2BE0B54475D0AE3E30CAE146306AA540E0D63AC325252DD66D5DF50AF7C1D3A651E5A14519"
01157     "2E1F05FD31AD2F37719B62DBBFDEB84EDFE0D64490",  // PRN 31
01158     "9DDBC6F5221BD70BDA886C59D62F7A95BEB41276DC3B5B505D883DCC478B697388421D5C21"
01159     "7C8DD2E56A4130A28BBD7176163980EB820899CF74F3074DD7E74A77B54677A35EEEAE93C3"
01160     "2D44EC4D2E4196366E40FD0C2A4741575ECFB8FC927142229E63FAEA97C51E12809FFD336B"
01161     "CFDDEA3FC03A1FE99EA685DAAB6F27DA6BACACD5D432352FC6FCC323EA9BC72FAF2569A7C6"
01162     "D55E1260DF0963EDD6CD3CCF97A033F74201E4390F64F06F89080C830986F8D6E90AFED3BE"
01163     "6A54202A59ECCEF101E5DB790E997918DDBFA1B160D472490F3CCC2B0E1C10AEC61ACC1F36"
01164     "BB6A8735AA86C50D2AC986721C003A2A8F1056BB9910F5A8D62D358DDB31589280BD14319C"
01165     "27A13D847D84C02A039B869B52669DDAD47CBEE6892E2A881A020867405FC1756DF2AAF2E5"
01166     "706A16F53E1E459BCDC430C26403F60B979A4D74F78CCD34E48321E67A8B896A162C873CFD"
01167     "EE8D58BE234C34FA20FAD472F0EB570F5B2BDABEDEA0D0483176163E20E47EBF51262B1944"
01168     "2016EE5867BD3345DAA7CADFB2BFBF14B4EAF986951B024D876150CD64B7B0A188CA79B122"
01169     "F274FFB39F1EF88D5B9D36437AE8BA43770DD69E3994F8CD673EC3E1F0A51E798271AA7793"
01170     "CACD7E53E27A325193A9B9FA33ADA3BDBDB9A916DC8BC6188C3D4C38848E3506E38621DB57"
01171     "C2B0FA80BE5CA34C295159163F23ADC249BFB1ACD94FF0A42054C1A2C304F3C515B8B44F2A"
01172     "B1A760E784B95D9FFA905EB5541D14D5F12B4489403645A15F004828AC8FD3D8B44C5C070E"
01173     "8A64BA086462E5CD447C58E0D96130E9BA3400CD7AC266AF3E3D33DF214D3E98C6B7585C55"
01174     "29797738879552FC758F04B122889E81B887B653FD7D4A8A5CCECFB0850B58CFD4BE990FAE2"
01175     "23803EAE972D0C65A97960421D3FF5B2B75079AEB42A92E0D0EC2F04967C91CCB84FE6E39B"
01176     "6B136CB64424686DC6CDC4AC39C26BFCE81478AB6429D4E9B4877EFFC8740FCB5D9968ACF8"
01177     "D69C771E9E48CB542BA5BBF0327A4C55CC83E466D6DE0E639A54576D85A44F305958A325CE"
01178     "98A45E2C227AA9E85BED652F3EA5CD31AB99E299930F5AE92B12E10D597F69272ACC8185C1"
01179     "E83412A67CCA39DB06A77329475A3A87A1D4586B191E2E8E7C3EC39171CE6E775E7B369C8E"
01180     "ED0B2E74A7233CDB5905F6AAC099820A227B6FA0FE5057CDCE6C8175F4CF27D70EE76CC0D7"
01181     "69A324AD66D013A3F699C7F024C22C03B353177C3D721EC56E472AD6B383A653FFCD078774"
01182     "C728258146BD8D928B209C0A6A8CC60069F160D33F673C139534362B58E065F7D1176FE95F"
01183     "4A6DB7CC75FAA408C4FB09B7026DD640BB08945B364646D9154D00CDF5BBE4323906607904B"
01184     "C3D70A0207F696152CE6FC5A0BD79625E437197B929203CF028384558071B625E0F8801EC1"
01185     "B13F6DF17EB8135687F08071DFC3BB06C8EB4ED296C821CFDDB7F2779B0B913E623ACA6E3C"
01186     "5FF023C58EAAB547BF1A67EA020F317658E1328FC845F65464D2E06ABB63F678A94614E3EF"
01187     "E281E41ECD6C85FE10ACE427FB1960BDF06630B20CF0401F450D71D3FCF0C477682FE342F5"
01188     "511DE926AA24E776CB9C798058A7576E1F76ABFD2C86B8C741230D6D95ADE980ADEA22547B"
01189     "88E607AAA8C3462FA34114DFA46046848CB10482314D1240516AA7EC7160215A54F69EBC39"
01190     "FE4CC97EF710BB7E8B55F9AC158B7F9C87B738FD993D2F0E35E4E9969FE5FC4B301BFCBA8A"
01191     "28D7150602ABE7635A2FB48AC65EA29B829DAE037D8711127F80CE85BE003B4F94E958E493"
01192     "D7C2236DF970D7DFE8495CE083E79B777D44F96C6C",  // PRN 32
01193     "33282734E9BBA791CE9CAA5CD9F80DEA9EDFCA4E3F5E32FE046FA978612D25F79FEC5DCD02"
01194     "6666C35CAF0B5A682C9FC6C97F54BEDC056A6D900DAD88CBCA846702B0BC435DDD5ECEF1D3"
01195     "356E85247940A3892CB61BC8FBFDAACBC1DCA9208DE593DEE6944822EC80F823B2ACEA3FFF"
01196     "66E4CD6F542CED77C81F29F144C9D0FCF3B61C4887C6B42EC97C83ED2C5DAF28CB8C9B352B"
01197     "37255F54835EEB978966A14AC91D58857CB7FED677EB87F8556B343BE0056744655B6D9761"
01198     "5F7F32BD4A537D8D586F477EF6B2871AF0524F71763748C57DE63D62D6824EC5075498D492"
01199     "A2C621FCDA2A5540038A50E6765ECD69CDD45D10153171DECB85AD45A8A6D3C609E350023C"
01200     "32E1F72B8D5E1AF803F68E288CEB2F5CEE3F10B7BCD7FB15D11DB3445266BDA913D5930776"
01201     "EEA4446337EC5A50F9E4CB230BA735B656EE26BD6BB5632B907CB81BB65EF4D85BA379E4C1"
01202     "64DE83CDCC023FD458606E09552303B8535B63E37FF038F2AAFDBBEFAC8BBBCF04EEB41BF1"
01203     "26C20C533DBBAD8FE10771F01BDD14AE113760DDE4C0D71FC7C29EB8D23D668E401B3A026C"
01204     "46913E2F08DF8434FF64B48036A37166F32CFC8747F13B656BA6D18300935195D0411E8B24"
01205     "E1F47298AE9F9617DAD3F39C4251566CD92A64860949BFEC05EE2EDD7F2DA0FFBF3D3C9F5A"
01206     "349130B7A5BC5B839000431D00FFEDB344C0DF86073F73C6833C091AE132D121F4568D011B"
01207     "A0DDC27DFDA5284478F5AFF037D89DD8EB9FB0EC066AF20740DAD8D29C1866DA0998E580BE"
01208     "BD311BF6E3BBB31D89D2DB162A680D2AFAC4903F9838B616EEBAB56C2D0386C65CFCA946D3"
01209     "3073684FA119B2A6411E1786C723A466C7070E58248ECBE1D86C6D6CAA199944D9944BD6E8"
01210     "B108060B612F8FE724742DC158BC1E5E3F71AAEF1864A7989CFC68CB2396D010E1485C3238"
01211     "F6385D68E20D36A4D3BB8A5A265E686DF558A73020588EA16B4B0D3F74916038A9954F7523"
01212     "2087822BBD8831222F7B94968C5A324018192A6377661AD1E15FF05AFFF75FF60B60C5B41C"
01213     "08F001D85A237443EDF3E361030FEC44573B044CE66F7A036EA78EB83E1863259BA363D517"
01214     "049B13F61956740B1DA3CCB5C9EDF190C89A2FE908916F9DF9B2BDF0DB50D032FAB059309F"
01215     "3DAFE1723D8031C93F9CF657F17D1B214492786F0D297D3C4C01A25DF2B1C3892B6724DC7C"
01216     "CA704E9AF0163D37F91DEE0850D9F2AD4860BBD2266FD7EE4E3D56349B90155BD843EBC089"
01217     "9FAD39C67050B5F8A1A9DB38EE821EF7561FCDCD6F833DD3C9FF3C5D978BF7E70288BBF665"
01218     "F4F20260A74DD0D414B3538FCA93671DB4371F2CF6AD94070014BCD5ABAC89157836CBA9A3"
01219     "4FC8D84090A3BACD7D641C5FF58A5E1489E1D2F410E65650AA30D7FEE4AC3AB37D5AFE208E"
01220     "F3023E88D96F4760120DB5FC705A005976BB27F580CF5C10988BD2221507F8A6BF84B19CC3"
01221     "BAA186050870D88C6B14F98E56CC470291FC96AC5BE10704CAFB10CCCDD8E3EEC9D58E448B"
01222     "C655A0A899C2FB39073CF28252FE3FBC6BB61189B154EEAA37A797A8E6F329C64DE5836EA4"
01223     "4F7FE04837E343963FE89B0624779212F1E5E399427ACC8BA83C10F3893C7551FFFDB62BE8"
01224     "93DCEE02AC893928E81A080CC2EC8B95C4FEF1A6B65723408F12F6E3B7B384B149F4002135"
01225     "EF114B67AA3CCD406A78A732374CD7C72814BCA01F4D44909E6CA1245B25E27A67F2350D28"
01226     "5E3B37218495CDFBB9C7FBF2C40FE31C57702519B5DF1D41D85F48DE4E46639104445FC2C0"
01227     "78CFC853B25BDC3E4C58222B2F2C5AB40F5906C314",  // PRN 33
01228     "6E2FCAD2FBC021102FBD4F992C597E4B91D2AC56A22E8FCA938D9C74533912782F75670F31"
01229     "BADFEA83FB93A0F6D449092FCB9236F50446A4655586FBEF31323F1795AC3216315BC9F4C9"
01230     "2FF583AF7E977A996F10989067975A53E153364B59662B6CE9E1EB65D6D69B9EFEFE57ADD5"
01231     "8A7921506C2BCF4D51A01F79A67D41436B519D314D1953C6D133305AC06A45428C0D6FD922"
01232     "8B7B9228C3A2C03EBCFAAA09B9343C736BF99022D1CA0B5C25A6179EC0ADD8AC95E8D6C00D"
01233     "6273AEC61E1196741DA2ED9436F353919A8D25BFB8234DA2229CCF0FDC4C65CC6C0374A4"
01234     "ABB918E1B5BC7B929F6554C568DCEA62D5289E45284DFA975D5AF56609EF4DCB3CA54A26F1"
01235     "00E00D48446C4C35D1D911CE7EBECC58347940BB411F475ED18D8A1315C069C63E59CE2F69"
```

```
01236     "E6F7B48B005107A69EBD9715303563F6A217E2ED4CD373648DA437E10754CC563679093094"
01237     "7650E6C5D1B3D1AB853A52A89D208301C54F8E7DA8D419559555859AD1B4D096A1805DA251"
01238     "B6767DC2C0D917919E119E71CC78A884A24F5061CA1A999DA65AED69CA02C20B6EB163B2CF"
01239     "435135D3B34FE6EF52F3AB84DE084E37958BAA4EFCE8165E2F5AAD1D84E9F7669FEFB6E60F"
01240     "1B9F69C79DE9296B7A01BDB04CAAE8E6A16ACF9A119A8E577919CB22B58EEB7EB289E708F2"
01241     "29ADF81E2350B211AB48043A491D4CEF1721936B71145F07A40947D2A89A584D5D0A708F34"
01242     "80F0892E331B2B73D8B7D88B7E85F73DBE5346D801B1904CD41A8070A0DC74073217AC4156"
01243     "54F4170F42314AC0AF4213CBD28094F8C4FFAC06BB0114B2C1924A4A1B7655965468574845"
01244     "864ACE82B8839036DEF45E88F341A5DA37A511D583C51DDBDD51577488A29D7663CCA7AB2A"
01245     "EBB53A4EEE7F38B8637C96CDB86B23092FE0CF669858C5374F6F078FCE4E945FED340903FF"
01246     "26A6AD6C408E390F08ADBFCA9646CF224A5410D0ACE93380314541982A8AB6443FC358B66B"
01247     "027A15722A0BB7C61484CA65A62E4496053C947D5F4744B69B4A73F7801453FE2055722487"
01248     "8B8FBD318E8B572911D09839FABFC267416B4E2245FEEA6967677C1C57A3F72E76BC297CBA"
01249     "73AA729B890E90A4B29CCAE4DF5D959BEE30D836C859F4F67A06B4A479954CF20C7D0639D6"
01250     "80EB61A2CA11B007EC9772293EB8E6525A56CC0F380ED4F088D8CADB0B95BDCBC84C0C746D"
01251     "4DE3307AC7050FA2531FD24164D29294D492A07024FDF1B988060FD766128120226E63F497"
01252     "3278DC9F07FD5699A1313002CC9FA7CB44A72C9C9D740AEF0984D7D4C5749F9071138BF93B"
01253     "CC6DCB601CEF5F60E4BF2361A083D158BE93134DBC1D4FC79FB866B0B215E230713137EB6B"
01254     "FF388307CF2F07CB07E974C8A874EB20B06C843579921ED0839F9454DCAF71BF56387E01F3"
01255     "DDCDBDE6C15C58BEEBC137B673F122179100613D6708B6B16AB750A3A311AC85E795FDBA00"
01256     "2CFB2A9D9734FEE1922D984DA9F76EF65EBC9F647B280B0433A3C2E73B08A3669C78CB9885"
01257     "F65E97A6B4D64D2B3A29184EA87163B06F267FE94714977525AC061BC54F7FD95CC0C4285E"
01258     "45D75772AA3543AC00EAA32626CE2F9DC48862657C1E156E08136F3B1EECAD329072EE4EDA"
01259     "B16F385D736AB806157409496A9CBFEEB3F5AFC91E2FC5569DA2C5DF20FFDEDEB6BB57D18A"
01260     "EF1DF03C7D5A16BACAD645772C4C016D147EC11EE9996D2E791C5251CA8FB7719F2093DD2E"
01261     "4505B7BF320204556D06B91D547C9C6B0B19FB57608934D4A4F748FD07F3F57CB108A31031"
01262     "9F7835C082FAED651ECA42EF823AD83D6B22375B20",  // PRN 34
01263     "22C6D5CF65DCB08CFEEF580D53CA6571787AA81CDD29C4918F5EFAF89E285A37DA51E01500"
01264     "BA0192D2F45573533F540BB1569E27A8865FF94923B81AC26FAC1DD03DF79DB5B5064F84A2"
01265     "B15E0BE7CC53BA875E477159D0352CBFC374B07573FEC884A699E93497514E7375E49FF5D9"
01266     "D9DEFED637DB89158F30B27E802EEE4E806FD61A46F8F7D7AFF89C9101A9375F73A2164C18"
01267     "B0109CBA1CADD42E13C9170378E28382EDFD85B80A02469D9A78637494BC6DD284557ABCD5"
01268     "756225EE3B26C7BCD597E9744FA1674C5088D3301D779B1C3A922A0723AE1BB253EF2BFFFC"
01269     "99BC8BB4CE539BF6D2D0951F6C76BB001B48E52387C95E49FF1A9D1DC12FAAA2B967754113"
01270     "FE59822A1E5C56A6053F73A2CAFDB0CA0B58312ED54B4F9D1832FA089C5923133F47EAB56C"
01271     "56EB7AF20D8006EF8D9674089CC49D37E82A1FFB88C97C3267569B5AD1A57CD67A0C94219A"
01272     "AACEEF99A11B28D01EA4329568E42A7E717437B030D3B649DAB105C700F30AEEA7B82A9759"
01273     "ED54F890C6E9B2C9F3F1D1A7A14A245817DDD4E2343BFDC5DE774EC60C5C41362805281C30"
01274     "19CB2915182F525576B3BF537FDD5236300DBFFFBCF8A8C9489B1A119D66181A79AE93E7DA"
01275     "8AB81A1A3E8D5A888F5144AE0DEA33ED8B7B6392B789EF9C544F876D179DDD822E3906F76F"
01276     "BD579FBE27B4A2752D4878962008B7873765B9BDD3FA581FA192C0C422C15153D7FF16F273"
01277     "2EF68B66042B5053E458518EB65C66E3955CCE4AA212ED4980411D3E4BDC6918B30BBC400A"
01278     "8A338DB982CF55F2C3B0C3DC6D64DF10A8CE1BC117338DA3987965A5EAB6C2E8E368F59C7D"
01279     "9C77750717B4440AADFF57D0EADFEE7C1A92EE72283D425BDB0CFA9F3BCF602634BD8526AA"
01280     "80E0247BD0A043D277A3E575C1D1EBDC0565330FCB30C63AD02F48D485AA43358D957ECBC6"
01281     "991D0358AFA953A41C67019E96576BF88122CA76D2804B73EABB88007A121BED76F33AE0E0"
01282     "433EA74FB2C7AD672D0DF5927D87BABFB30989EB91BBBDDCA40ED8BCAF7684E8EC9D1017D6"
01283     "147039D3C3423A9A0081A17B70F56FA300E8F2F97C399498F4AE85FF2707B50E3A4492BBEF"
01284     "19E14834B261F700DF4E685DEFF476ADD919D5A15A7BCA06F382E1F0649B6934D70A80416F"
01285     "732552872BEA44A5A42D64844532DCE8ACD5D20039D397BA9EBE04EC4742F4E2DCAD29F89E"
01286     "0A9FF572579237863890DD13E7168CF2B4CA8F6A36B49A5734BC85F811087B9541605F44FB"
01287     "EB3313D3B56EA13676CB69F4FCF14A1BD285455BA516F20B2B0D3E8922B71C0731F2BFD112"
01288     "4067D4CEF2111FB754CEE5AB73DF629797FB5E012515609B1E9A2085643691BAE5E4A2A52E"
01289     "904651DDB340090EEF578675B778D1E6EA868FC1106F19B0435BB464887DA311545300841E"
01290     "10441E9316656698C879089FD009D3F8F49097B6AB6AD2D744AE79F1D15F0DADCA70C11773"
01291     "74ACF5D07979CED4F3173FA4ABF571F6DF2112E8AB26C257D6861923CD92CBDE33683F27D2"
01292     "CD542ADC623230470FF7E950475EDB761E57ACEE78318FD60325B871936A51FE23A63C9B4B"
01293     "7DD8A72556311877DD6A8EE053D6CA28261BD359C0C997F2A184270847A13FD3C952741C5D"
01294     "1C47ECE09DEC3D8E0223110A627084BF826548E60886D7EA15C585F5A16A813F030EF3D150"
01295     "FE7A4D271856623D96742E848F7797526238E56C102191600AF7756C562A9B627B1F05B56F"
01296     "B2C768F6B57986BDBA95816E0842759438E262FD9BEDAFCB594731E9A04512A22D1EFFE104"
01297     "EB8F7E502B94FD846C71004E534FEF2A30812F84A8",  // PRN 35
01298     "E881D91D5F8EBA0700605336E6BCA6A1B1BD1D58F39650CD55F056FC8360EC9A2CFD0CF440"
01299     "EC2C6E98EE88E6B31558B838EC9D473F75CE2FB04155330C9FEE2C381F872C3A79331C536A"
01300     "05C766124E69AE1EBE9425387279FF9FDCD28CE6569215A65F678611C96103D710E4502952"
01301     "1D6EF7D5A2AFF378C9C26ABB52AF1ECBBF615561932791A23A3BFF2824CD3810EFE7143A79"
01302     "B4E14132E306C48D9B2FCAA475A7D2AA0BAD046314E5BB4839B27FF4ACB7F4B3CA8C97CE97"
01303     "6F11DF6DDC71144A600DBF9AD43B6E471791CCE636D67CED546A377D786CB085D0E4E91258"
01304     "719A9233F342180C9A827FBAF77C8B5C1BF72E9A81E4E4908EEAD39772DABCC8CC0BEEF081"
01305     "2DEDCDEDD7B4110C6A0B89484C2F0F6CBAEDC219BEBC85903CD03AE727B1849735F62B385F"
01306     "4D5CB58C527CACDB3586AF4525BE1CE424A0879546505F86E324EFC4FBB0B9E86EC2434328"
01307     "66DC9CCD4D9FB258886EA75AB006D87CA5C10B5BD0E0CC4DFDC6B152D65D5FB268DB6AEAEF"
01308     "C9EDB4954CCA7E88DE2269533D54AAC447124EFA9BA62FACD2FA914885C162D59BC6BA2FAA"
01309     "3389DB9AEA4C350843B3361C9F343955AA616E661ED1A137EC0BF957BD7DD7105856499367"
01310     "E91849C99F0F7138CC7AE120CA36164658BF3B67BCFC1BECDFFF49606448AC8B38619EF4A9"
01311     "97AAA688D3FA7FF169B8BF537314CA8ED865F395E300E5AEC2CBCBC5A8B5AAA31C811E3968"
01312     "FCE32B82CE5EB1EE839134EFCA68E4CC4C2957329A7CAEF675AE2F990CB108CAE8EA5EC21F"
01313     "6C442CE1B818ABC203327087146D67275300868B95052261AFCCEA3EE854F38DE7CE1FA564"
01314     "9AD7E9083DF2DFB3ED4186244045390F0C843777A677B8A82B69EBF43D8FD24C9BC0792A23"
01315     "36581F3B45EA6F8BBBAC89A0314570084D683C76D32C2B0A6868A428DA7D72D9C6D37D99BC"
01316     "05D14ECE27C6C1E3F849332C1CD67381B6CCC907E1A911D3C97F1B40CC11822257491AC9C2"
01317     "2642C4FA2CF8754E6AB2574AFEC65062ED5C1E914F8DADD52ABD7A9D74E545C362880F38A5"
01318     "67BC3E6D7BD7156C87F698C870008C2AFC78F94354C2DB9833A0635E2C63E5A47A828610E2"
01319     "A5E8B2F914306C966D1EF8222BD8FDA2DCADA05991ABADAE250B942F8FC82AAC026696D62B"
01320     "FB27C8731149C49114EFD2AC47250E0454E33383082E7826569AA4D6F5661A9968C1A784E0"
01321     "A080D6BA7F8956E37AA55C07257AB5971E03069E74D245C4A2A0BA0BA397376141FCDD486B"
01322     "2B3C865E97D1CC327E4E3320A131A91B61F5A78AEE77D1DA75A1F121D777DC4E3714B0B515"
```

```
01323     "DD96F477E9232472353FF7FF07B0C23D1C88F7DC4516C36B3D1820AEFFEE62EC3DFB614363"
01324     "0E1C532A4FC9740369617E80312BF3CDD7E3723F287DF666A34FEEE38B3A1461E1B5FA0390"
01325     "7125359025AC125D6E86FC6AAF0223F83F252AC0C6D578DDAA8A2841DC35AE6B6DAB2E7A31"
01326     "EF4282DF4E20D471A3A8B28EC6F510453728F981D7BC2F1BC2A8E024FA65BE18AE8CD272B0"
01327     "5403E05C6006AB1E612CBD9F8A71C50D618E9FE46B243AFF2529F2A0FB42C7B990BFBFCF6C"
01328     "4CFF59DE230EDD411BC93C398C49E76FCE8EBA3C2C93752B38886162745ECF4F80EB59AA3F"
01329     "3660C4B1D4B9850632548A6D5A9F5A8DFB5A74881C0E93E38AD2FD1F0C23E66AE4631F25DD"
01330     "442E1A010D6A04FB73559C76CB30494352B24FEFB1E529BB6407BA7B68535D06C7207B2126"
01331     "A585BD59C43ED1E36AF6408E33EA1CD6F43BA41401048AE9C5638077C8AA8A033F0C18E589"
01332     "9A6E94005317314867746E30C1A6292FDA7D06A6A48",  // PRN 36
01333     "74C4DB6B3E8A4BDA889FF8264534407C16B05809DB70004C8EEEB10156C32686ACC5076D21"
01334     "A2F583EAAC92954996BFEFEB1A28DD699B55B5B94A122A1349C58DDFEEE9D4270368DEF85D"
01335     "86CCA4FFEA81880739A934BB8831AD75794986D60A92AA6DE661ABAB1010F3999A57A53F38"
01336     "68026C64307C47370E0B82FCF8C02A3155E787DE359624510D5008E229E9DAD0005CFC9DFD"
01337     "BE5080BFD01D7342E570360E411FC27146147EE2C72931D0577CE6D7183386C754A6820BA9"
01338     "7BDF080F6EBD063934E13B005CADA4D2D8492E143402C9D1BF8A04077C4BC7314532474704"
01339     "BEF9D24E4566A1409F085C76B65158E4EF6B30141DB42B0A96454E451BD63117428273D362"
01340     "9E2E5FDE4D9E53FEE5F9EA2F111A0FE5F50D2B72DD26E941A57213EED915147473D630F755"
01341     "6CA46FF8EF1F0C88E6272D6E95FD3741AA675BB0ED839BDE162F9A308B3B092663B1963221"
01342     "708404CE8BB54F61BE9AE987344228BB60921726D90FCC07D5F64BEF673E06B969511E111F"
01343     "0293BC5E57186B283AE81CB2803363A1266E7A78B47F5A35AAC2FF0B3A348DE73C81D74E78"
01344     "68683918FFAA42A91B89E1E2AFF4BBE6F11867DE292A6FAAA689CFF87F4A629FC354ABA202"
01345     "EDBEA3F08609D1A4C350A7BB7376A897AC15A1B0CFEA0D35DA71039A97B8C99A5367DE46C3"
01346     "38D75A845A4CC7CA292DF57F2AD8C5E20D95670E3748E8A1BBD3D728980DED4FE0DEC90884"
01347     "B7A528D0B4C27DA315195B7D8D8CCAFEFEA3D62AE34B40F5049D06C3AC92CC59A85C5D5BD0"
01348     "4DFE0CC89A7DD7A1B6AC4F5F3D2AD8858B83C761485B77F6D512DC2EE1DA402FC6158C644F"
01349     "14C2F6274E8113B453F9130BB6C24E36BF84C2942F8B28A3914110BB3D48DCAA3ACB1D7A89"
01350     "769E0B9048566671760AD83BA7F7209832AC10FDDC299F7F90A3744F72ABFBAC19382EDA24"
01351     "D0BD5859634EC3790C7390AC2866B1379ECEDED50D5127324A4EE6AF50D9A6826DFB3E27B9"
01352     "922E4A8652526A6DA29A4C2B9B8465E3F99E72A82A27645CDEFF1C42E77359FC913A907451"
01353     "6720BC61EB171AF90AD427A574080970282539A05471AB7AE2EB027DFD3D496BD27746D0AB"
01354     "A30EBD9D13AD9DB6D405CA19EEB90CB088D9CFF93B20DE07F86B978C7FEBA8AC29FD4E257D"
01355     "23380B955CD79E93E3135BD14AA338FCA16560A123131DDBA8A2B63923860EC3EFC2E03390"
01356     "15662C3DF1E837C879969573E0DC4A63EB7C0D6E7419F079D4CD05A1A9791330F1F1B7EBAD"
01357     "8CE76933D54A224571796A3A046116964B98A7EB5D9FD8311CDD6BCC9B63F2B7302347C0D3"
01358     "407219228BBA73B8B53DC3645EDE1969D345D8C23E691A58370B351BD6C86EA6EC9349D524"
01359     "BD05D07E68F853FD9214414507ED507F0E3CED126F95630FC6A3B40152A318278B8605F30C"
01360     "389C1826478635753FCFA9D227325463581DC51EF4AC47E1C49EA005D5AAB9138706671D64"
01361     "082F6E2BC39A802685BD8EE8001031F7C56CBE5FC68CA5B14109249C210B4ED7B9B2ABB4D4"
01362     "E83C9EC82911E9CBCE5DFAC8D4A1A6CC81B5378D29E8E339CCB31B85D9DEE3080ED75057CD"
01363     "5EEDBEDAAE7F3CAB4BB1BFF54042E28C8CC79741FB04246ED66B3E34D83F08D98D7070D3D6"
01364     "A032BA442E46CE38F45957D7485B17B51C562B94985FAEC8362A8D6DE43FC0EFE1529E6628"
01365     "99F1D21BFB643C22293E0A0F2E1D40EAC1934A0421C1617DDEC3D40A3AC9EFAF0F8D94FB7D"
01366     "EA085C8179290FB4C917C7D6F8B75DA8790404F8191C0EEFCFF9DA2208E7E5D382B5A9D1E7"
01367     "2089B3C4976036AD530BFD31F6D08C2CDE5C26E68C",  // PRN 37
01368     "13AB0304EA5EC4D96147EB5EF8AEB8FDEEE70A62E36686ACDC275D2FCA1ECFCAEFDC4F8CBB"
01369     "E5BFC2BBD343742EE011379103102BC11E824732D147FF93CA6C9B0A503C49BB2B621B47D4"
01370     "C5971643EFC01E760EE958A9764397AA551423EFD49AEC50E22FBD9AE99659E36341361286"
01371     "D8452EF70C662DBB3A421898FB34EDA3314A1C7A6AA9E9B5A6960156A549C32B91246A9CDD"
01372     "CCFE0FF278C3285A5F20081EE6FE121E3CAF922C75915DA773D0673EEBBB268998861C3018"
01373     "642A6A1B1FA1DD8DD54991254067B4E9BB95D1AE807B8A41B88B772B5590FA9F0D71093854"
01374     "C213283093D5EB9D4C7B1D57DEB27B1E2A1F0CEA306B28A5DD3D1B23EBB7494AF61F771052"
01375     "798C6FA104940EAA5698AC24910DF692D5C89C29F318E090F30CB12397B88702418C467EAD"
01376     "9599670DA2F993D5B6FD7A9A43CB89274719A2F1491FC3C3E1528EC00A6D3117C5D2E0CBCE"
01377     "BB7B91881CB7540809B5B9BD4DD37BB6B941335049818294 10B4F348A5A7DC553F089C5F75"
01378     "9F032F0F45C13991C1F442F2DAA5C43C5A6815383091ACA7101735A0A3D8E5E88A454067F8"
01379     "D92C4090DA1035B7880A303125ED48A5E4EBDD9308BBD45B77BA6B84093210D9B011E3FC53"
01380     "13F2CF1B20A9C08D3F4D8AD7003B32A49850808F1B78115D1E727A7BD387D2D4C4CBAA4CB9"
01381     "4CE71FF25229B37BF1D671CA50A04EFCACFBD980263EF16015A97909BAC6FE4F981B96AFA2"
01382     "692CF3603C5C655BBF7AD4DCB06CC3BBF8BC819CCF20619535D399BDB5FD0D40CB261FEFCC"
01383     "D4B0B0FF3684FB0D992ADF1AC6046BAA8C8E0F16B6E691A23DFF7014AAA67C7F84A9429064"
01384     "139809E1569145DF3CD8240D49BB9D9FEFC2176C697D9950F11D56979DDC01ED19EBC96908"
01385     "93D6D8E76041BC4FE35982D5C535FBB883654FA3A2253D52F06B62022744442D880EF9DD4C"
01386     "C2141DD25ACF2EE3B26103D6A16710D8BDCD43731A59DAADC1B40092D9B166E822A8A7AE0B"
01387     "4323B8CEAED7AC3BF3CA6F24BD79A58DDFFB59FC2E5FF634C83F6F7B03441DF9B5763FF827"
01388     "D4E74764C3E0EEA21ABC1AFEDFC2BB9F73E3FBA165A7DF3ED9997A588CFFEA8E3C193F505C"
01389     "741C1863C3437A80B41B7668D12B0DC7CDC6A61923C102D3D9071D9937F36A39484ACF1A24"
01390     "97F238E78A9D8C9C1B96CAB0AA15B643B81F6C5439C2336055C8DE4A392B8C110753E060CE"
01391     "92325C3B67F0610D79D46F2E7FDA96538D43B09DED1D647E8289986EDFF482FF7E24F40FD4"
01392     "89E4051F7D0C9267E934ED16B13827A8C3D47C192F5C5FCF6A8E82521B7062F6A23B463262"
01393     "E425F4820AAAD8E23E85DF0448B00533FA29FCB815BC83AD29B32D12F614AF4A6C60F7F133"
01394     "9E639706C4C671B71C99B55056832544A23C996C6BE5230AB5D8AA6B89A0E05B4235CEB352"
01395     "C70D3787F740EA6B8F05CD97409EEBDC9F1F98E428AECCB124F656EBC316664376CA4B3804"
01396     "6E5640B377457EB82A5CF0E81966865908DB4B1E9B482F8740A470E4F5F0333A11E1D67A35"
01397     "0ED7BD9C8306186E83FDF6FEA9A09BA1AE266A24AB6D3C192A4DF53D0A627EA2A2847AAE9D"
01398     "5EAB2DB183A45632B2145C3A2FB9C4EDCEA521E4DFC5AD34FB6E2CFC2E5C38B30996542C09"
01399     "CE9542E2F692B863CFAAB27FB549BA99A1DD77CFAB7E26057055737E6D08E5592D17DA8322"
01400     "B0DB650AFD6ACF4ECE23F1C9613AA7670302428512238E608B6AF8E0862452319E3ADE7199"
01401     "28AFD059A58D39EA819F3C14DEA608F80D3C93E374C2170D2836E2EE9F600A010B97418B52"
01402     "CEA1592F4D6C4E91547EF019AF4BB4F1ED010242DC",  // PRN 38
01403     "1193230653E34BEC0CC617609C775D5DC198C2F533BEE5C3537F381DE8AD1995303FCEA246"
01404     "8BBE8713A3770F59EB4F58397BF63DD3AEB17BA59D1D906C853B6BB55030F2E0FC04C0BD74"
01405     "6B755358EF16B44A6494E65FA26A294D25E7567E7ECFE7A830896D9CF2A8D97DD5D6FFBC52"
01406     "1907C1677F195AE7539C564DAD1D78DF994A3F33DF3F0400EA6AD3A742FE3EB8A1855BC78F"
01407     "5E9ECB5F6E41A13E2411902BA48D0061D318C1D026FD379C5E4D9ECC3FC5AFDF0ED8D99E35"
01408     "FDBAAE7DBC9E228411D203FE2645169B3E8144F508F88D380EF8F9D6B6B7696CCB9041BC14"
01409     "5BE25FBE39017DE8120DC3989669DE5B587E5F2C5C057ABEF2195F5E0AA441E05E256B5BDF"
```

```
01410     "CF6363507777392CC1D5B5BCDD59C82BF8F3F885C86F5E6ACED0F0BA503944A9E2F15E8ECFF"
01411     "BC5B2C1444776A5FA7DC54EA0B1C9739E21E2ACD85DF739DD5761C17D71444039B3CD53EA4"
01412     "7D67E8E8C505C39690D09E52127C8F055521EE8BE8ADC6479EC436B96DE2CF8784C71B585B"
01413     "1EC1475F16D92DAE9C29BE7375C527B1CFCEAC5FAEC37C3D892525FE1ADAA164E1E90680ED"
01414     "F0876A0C50CEC8123CF2EE2CAE3C35409ED5C5E6840E004C9F69918AC8B39899D0B846590F"
01415     "17243210FB32D3CBD4E3F8AB4C7C42DACB17716CC5B330B009EABEB31FDBE086602F370EE9"
01416     "172D66A8B4991CC56A0AEBAA3A623BB37EFFDDBE273DCA7B0C09649B1D0CB5F811F6A27ED3"
01417     "370D1A1E922A70214AD71C2668077E59C562DEED43C282333BADD38CF3B01E3C1F8A55B382"
01418     "2A7D41991CF4E4ED36D2D7AD71BF6BF100643EDDDD23B6C04486DF1193F552EFE21BD9B5A6"
01419     "A1D93D4459CB14764705290659953 3AF69D3CF1ACB1EB89EAF3004B2E4CCCF6D4CE8E3D6CB"
01420     "F8BD90E6D4637335D5939A3CCB8CBF6F6423388FDCC9B11A159813FABBA39263F337B3856B"
01421     "0A6B3102D920AB5D216FF4D370C836A85B753EA0E43E2D9E9442C9A4E64A061A0BE7CA61B6"
01422     "7B0F012CC43E42ACD298B3057F9CD3089C59B39200E8B005772AAE2C2138172DAE25B7D54F"
01423     "CE9C6F8A410C50079012F069A9A7D6581CB6B2A3EA9E09E4F2C1C57BC9F127B4EF8E392203"
01424     "71EEFEF0254877B19B37E0785583683E5AC8062B438889130038ACD1F52098E5E4AEFC3717"
01425     "016321C263B4155BE8418F22E523F514B1F8BE737449A48EDB9F3B38D55E9C26DFD4C8BCA1"
01426     "11D162DE23984DF208D5AB1B82623D2A4F4E3A104E3F84F65A020BE96711BA3B3BCE7E6630"
01427     "F8336B16EAC987E8FF10D97C62E9B2B1B10AE466451FB70D345A88A549967C367F6DC1C87D"
01428     "817EB7B4B0A5032E1A72D2078AF3CEB8116B1BFA0BA64F9C1352577BD9B9D0C552084898EE52"
01429     "AC1613617DB42209B5A8406DA72EBA89CBB1893E69BCB0E0058E4C227ECC507108152C248A"
01430     "CD033DB3C276E77D80AA7FB7C7D3ED6DA7CA79D469B1F2668DF1278FDDFD20CD02668C429C"
01431     "CB2D64E6AE7EA6C65A52B49D2A18EB48CDB26861AA872D74D51BBF34C56F5455C98E02AEBA"
01432     "0D3E658FFBBB82960205A1ACC2988D274E74A3C885F1E1F0660EEC75EBF839E8E3D5057F7C"
01433     "39DD8369F9BD6EC41A918BC8A42291384E2688F20A7B78A96746263991D57BE8BB2A516D9F"
01434     "BFA2028EE46CCA591086EFDA8C97B2DF9CD939D99A87A0AF0AAC65ED40B6551E6F84E69948"
01435     "3CEF2C9099693B895C475C52C56598AAE7A9848CA6AF2E7970FB80768704209F117776AFC2"
01436     "30158067B52AD83AEEE587F81014B0947A183F629D319FAD66FAC270A4D115F46983309789"
01437     "7A40F1856870848 16F1D2CE615FFCDB27D060331F0",  // PRN 39
01438     "594886432182182A60A8457BB3531920208BCA5F9B84F988DEC9D71119DEFE0956740D55AF"
01439     "CF2FF33EF9801B2C9EA25E0565F587553A9DB639F3843EFED712A9FE49D67C197615E19304"
01440     "23A24D464A113C76A8545601964EBB9AF89E236A2021D25D5A4F71E592716BEA4B4F9A9BBD"
01441     "46A835D49379873CF2B0129E236DBD2E510F2CE164F08307C4EB11CF12AD26AC9ED8863B0E"
01442     "F1FE28D2FBE71859BA4340B13DBF103F6A1A855C4D51717B4B6435E6A34F7122AB6C214028"
01443     "766BD398C6A03AF19D67D7E9CE96EC95FF00CCB56851289D52FC2F2BBD6CF27F8D7BEFCE68"
01444     "17A52FC06A468C0BCCC6E27581F11A9394226D7EF8E1E7FCA24295275042F14FE0BEB7C1BF"
01445     "A5A1161FA57796F21C5860D6CC3C9F2D28F5C3EB0F18D27D9C512A4C7D3A3C8B55E0452405"
01446     "643091F679A9352E2B3EB74C0562DD08A45339B1517339C19B88D1E6B13E341C7B932D3613"
01447     "CADCE8B349EFF25DFB15B9306B0520006C72C01B8E1F5D625C76778DC8ED95D62DBCC65550"
01448     "0EEFE96F90C8F5D83E4A9FBD2BE87594B6DE62DFFBC056824A4E4EB61B6E83AB9BDFCC9804"
01449     "FE3E2EDBB788BF3E421E62644BCF44B8023213A8FF374A14B8A1EEA61830771C6E2B48AE0A"
01450     "C68B14601D8AAE182A840F98198A8BECF4EC119EECFC5D7FB4D2920AFDB10968F70E65D2C8"
01451     "D5C8102DF73BF6AB44BDAD9B8577FAE45C83D76A3C6E425664131FB3F38E5946B47C0C9444"
01452     "416877D9FD7F3923C5EECAFC435FC17A806C13D79055844164A8216A605615312EA5792592"
01453     "AED524608BFA44DF569B5C6DA880DAB9C2B7CC0171C785ED77104E08B61C7B7625E80EF034"
01454     "EEEF7E13502CC6510A8EBEE8D7C2D6DD85A6DF2D13999042C0B3E78C0188A741B399E212FE"
01455     "05053B9F166EF9B876B17AB413072CE8DCC6004B99233B73F1E9C494B23457767C2197B101"
01456     "D9BAA41BABFE33224B2C1C112E1C3C5F099320603968AA648B2792E7D0F009C1C7C33D6CB5"
01457     "B94CCAA72F7CA5EECC827F34457DA961B39DD98A64C0DF1A1618DBA84681657E108C862072"
01458     "3F31F04728814055924B638319F13011EBDA0289A0BEEB732ACBCB71FDC411B9F981969023"
01459     "F6A3F8411D9D9681DA28DEAFFC84195A442340A537244E69989DD7F5A0F9A33D9ADEF27362"
01460     "321D4CD3A6DD3624D98E8DD1709C3AC48167D91CAC26705A549CD2E0C62621BC275530EBF3"
01461     "7AE97458C3CA841E264976F2CE212462AED47D5A0F6EFB27BB657D47BF6BA303CD66FBB1C2"
01462     "73A83EC36A4A2EFE2AAE0CE1FEC1B8740E83A73DF7A3B24CA22BBFE52584824445B2ED95D8"
01463     "B8A0BFF28DA14D382B8B54ADF00AD82B3CD5ABBD4935738679DE633243B4C6A6CFDAFC43C9"
01464     "08492CAC991FE4000124E0745CBFC21FCA91FB90E6F1183E5E96F67240DCF790D87A569F66"
01465     "03B185DFB5524BFC6959F717891FCDF39C7B66210BA5837A5645C62777D83C07125DEF9E63"
01466     "71991DA53E79A7B5767B6FA08450A806B2E2148B50F9FBC79FC66C47844E3CB6EA74C7DA86"
01467     "F313BA8C54F0FAC364F5C0D49F2C10238D44446AB58EA8FA3B5A5F14ADF3B9FF1AA225BACB"
01468     "62F3BD6028EB1066D7DF265BD210CB4EC7DE71CF028E354FF81B76E3C621ADDE054CBC1366"
01469     "FE069ED6F99BB74DE007B9E2BD31D1A96536F2260FC137AD11C5D2C2108C2C8F3C893F4A6C"
01470     "4A7FA2DE0AAC7931FB85C5882933C128A9C3191C35FE3CA022427B6AD9447A09B6EA0AA9AD"
01471     "70916E9127900DF0457C6B901FA2740353988E26E4B9BA1711E451F375029FED9AC0256433"
01472     "3761603E99A1B725136B5494538C88FA2357FD7D9C",  // PRN 40
01473     "9F4D899456FD3841B890A61EDAFD40C7B43FECF9D700EE269E58F40BF2CF47CC1C1EC7F28D"
01474     "78909A53C7B79C4E6AFD275BEA739A3617E8B3C3C77ABD4EB770B9E663BF154E77205ED1F2"
01475     "C5669CFCA18DA365B988499547B2C36D4A56582FC610E06E4D43E54CAB563B7E5DBD04CD04"
01476     "EA83DA72645F1F26422D416AA26CB2AE11C1EA270CF7E7AE3679FC2D69E012F0C950FC64A5"
01477     "6C7B766F909526EFA0AC383E4A28CC5C52A26CDFD2C3FD2AD820EAD3885DE63114DF5D19A2"
01478     "EA6C2BB338A2171C73A7C10AF41A9E4F0E49142B363AE9D306378B3211CA3B0F21EBA2E72B"
01479     "C131459E9D05CECFC76D6E540587073B0CEA1EE973B56EA2E671353210885EBF3A6465C1D2"
01480     "AD12C89217F1ACB1B209552F1DA1543293D6D1D273C4AC5AA5A4B9D2B368AA82A7FF8E1025"
01481     "9FD1FE52BA14F45A40F72B75AB7DA9F73DAEF5642CC1744E6550ED6CE2E547F33598034E89"
01482     "70A74D7B1D58FA2D4248186083BFBB8BC1A34C744B6F08AAC374C28B1C0F9E8C2B220C6EF0"
01483     "B5C0400D3B725005A711EE66FEBA353778ABD4552DEF9E5309FD0002FEAF1F3C5AA3753CFD"
01484     "7CFD62907CAD3B1F68071C01886748201050E39EB0DB18240C5D598D213CBC508B8F0EE770"
01485     "8CED659CE85BCA8BE714DA5A607E023F9609EA3D790501C4A9969FFCDC7B3E7317F6AA6B99"
01486     "CFC32FF1B74AFBD42A9773EB0615598C943BF55C1E8AA3B85D5D2EA2892CC33FA809D71621"
01487     "B02D141C74722397318B8CA0F4F9DEA01096234025573995DD2F915D35F7EBB6F7F3A9E83C"
01488     "2D23D769B639042FC669BE9778390A3E4404C04FC9991FA6860AC811BC07BCAE69756FE9DA"
01489     "B27B91B91C51892D61E94DB2B486C25D13C225810620023F1B9FED1E89B002B7B11A609FF3"
01490     "48867E1CFD827199A5F0D6FA267F60F89B09B5A1D8FF62744B386709990589D0557F5EA477"
01491     "974CBB87A86F2E6E94B758607454E3937A539C85569B5E68ACFE4911CB053CE207394B686B"
01492     "962A7487516971740287DE8C51CFD26A14AAA3D974068D227089093812DE910649512D966C"
01493     "B97FCDED09F4BCF7749E16E3391B75A4846F65EEAABC5A1CC89B1D9D52C541DC65CC468B27"
01494     "493F76F6D68AFFCFADAACF20BD979CD63A42EA3B8961EAA2F731C6250436A5A739D3E015FB"
01495     "32E07D373FF650CDCE417EA85A6DB8A9A24FDF62487855DA1C807943154C162AE17BA4E477"
01496     "531100431D647F2D8AE76DEF070A0418D2365549BC66DCA95722801F3A77E435205E919B4B"
```

```
01497     "349ABACDB2BD111087B05C05FF01E4BF04F5685665B361D37634855BBA312E96F2C6314590"
01498     "E298770C3CFC4207A0FC944622D4CD7280CDF5AA241232B6B5E78BBE5C6684319AF624124B"
01499     "E84CD2F1512E89D6CAAB620009206AAC993A9F4D6A67859C20A8B9307D8F8E370F2E600A9B"
01500     "363C2A460E0DF5AC007EE110F4C088CFBC2EE82D623D5EC42DB739C2267DFC81C830244A6D"
01501     "EFFD4966480D6EA8D258FC047500062F42446230456F9B5405610863500A8AE2440A00CD16"
01502     "E5B6136B0BF1611345CE7D70319962896B3F5E69583F8D305B760E0ACFFAD762DD06C54EBF"
01503     "8075C1B0ED97FEF59FBB48A7A65C2251DA406D5E9E5F67586C64E5F8FCC805DA4F33BB583D"
01504     "E2D5F85A348AFF0215233EF88C310FD1343934F5AA2A2AAF8F58DED34D0E35D210FF01AB41"
01505     "14E18019A2F7D6927FD843F842AD6AA0E171DDDC2BCF6B49EBF5D91C35665D8F88C0924A45"
01506     "3FE96CC06617D81DCE66A3C813F1D3DCD5C37E60BC672ABA16C2025C78E9A1A6B1CBC4C022"
01507     "49AACCF55862C538ECC42D0169B7E3F4B188F2698C",  // PRN 41
01508     "47A3C0CD552EDF50F80D3E5FE68A22F85EFBC41BEEB787E5CF0CBE3266EF310BA5E1B29A7F"
01509     "FFB0945121EFCFF811A1B7DD768D3ACFC21C4E6EA1EF06E851F2FB37A5F8E0C49F59A28A4C"
01510     "C2380A0C0B4043A7493320AF4E34A2A20E36A6BDD2AA49088FCFF24C84A4D53FD6D3B29178"
01511     "F921768F21C5827D64C125DE4F7F435C6BD000BCC1C95D0F82D61581049CDF2A34A27F2D9D"
01512     "DC409895B9742443D0CF74F8A7DB13CBAD88E1F1D6F52E4486809E20AAFBFD82B297B038BE"
01513     "774BA13649ED922FC67F6DB9F0864B742F6FB60E849F9A253E8B0B873C8501F72D3523EEF6"
01514     "2C1430350852FD17177203DD2BD4065984190EA2075B5385E1D29226236ED894DD10FF82F1"
01515     "DAF16AF55F0940415685B3A58D1DF7CE0B0F069F33E6FFA29FFE6134285CCCEA2DBDF1B901"
01516     "A700765449D67C24766A042D43695D1F5D241974E62817FF9259879813F37AB1E8E51F6C65"
01517     "4324118E411ABFBB6448E9EA3CF1F636D2142D65775EF9FE046B80F0DC1D1480158D603626"
01518     "F7002290058E62DA252328927BD20C05E5D3BB0CDFB6C60C04DD8BF4A3C9413241339AC186"
01519     "B054B34B24D9ADE710A79978158F1E2EA5AF6FAE9077CEFA4E335473D29BEEF00624FE6A9D"
01520     "EF0BA52464028BA25D9A9C1BC85901F507AF046D56B17CEE0F23433959CD7F108548169780"
01521     "8B99EE619BD0500B55945186A81B89A9C6B0E32F1D5196C4787BDBB08A6F1D79AF329B3830"
01522     "4B1203E9B8093146B15F1207C7EABB020B72B2B92EDB95377D698F10CE7737FA3DA501E0CD"
01523     "09F3FAE9504ACC414E63F60E2721E759CBF2CA1E2671A98656707F4629626219E4F16D8A76"
01524     "5AF3E75DEC1B6AA82622916E47872DF33689D820DBEE6EFF26E70F61ADB2EC3D8FC9A0AE30"
01525     "F12E3CC5E9A8E1A820DA15952401D3910F6F500D4CDB7D85401AA4C8F418F7E2A166B491D5"
01526     "CA3B415378891B359112894274F25E9DD2B391A96636D468290AF22FC6BD5C2E7842AC5A62"
01527     "BE24D80E6AF79C6F2118C126D60720601272FA89F966C812EB875052A0BCEC16D0E6C973EB"
01528     "B683214ADAC9BCD0541D32D33AFE16EC2BB424EC8BEAEF0BB8A3DDB18EFDF58EBD84B2DC8C"
01529     "7AD54B86E698BE762E73CEC7C1DFB290F6CC1B63F5257FBFECD9EC87D8DCD2FF393B1C1D1B"
01530     "B2EA9BC10323E6CA3EB8EDCFCCED055680109C03C5C668BB8848F38C4C3369A4E5C0515C72"
01531     "45B0C092F0B8525434468E775F40C4D4206B6420039BE1F695DE802DD718B9AE4FB41320C2"
01532     "20F6B9971A4CB8E67C6B0CA3049C2772D2B3C1024A502B48362CE9A1B84BF4C9D47D62409B"
01533     "5CD9BAF1E18DE57E9BA9CCE077266D720DEB23F192C70145FE201561AA2A46B55F9B6E5BC8"
01534     "FAF888FB74BF6510AA7E6BCC40DFD471D6CBD2FCC28B33EBC763E8CA58D5D74C1996D66CD8"
01535     "27A6A6B9F22AE7C24B13799A7D0A9C9DE940CA217FDB870A1640B55C69B2E704EA21262DA8"
01536     "31C7E23411D81773A965DF6AE015CFAA0D5DD3890D0327C514968C2F72A02A0A35243616AA"
01537     "D673DCF57A684E1C9A6B6F35E12C09553AEA89DD40CFF53EB16C5EB9F98B6D04674F8DE9B6"
01538     "AC125B8B0BBFBA71267A6BFBDE26B64FFAB35E5C5F27B7CA273B901D28997E6298E610404C"
01539     "07569E906AF09A968A9E40475EFEBE524357B044B964136C989FFFC70244DAEC0CE3BBAC48"
01540     "B5A0CC79F20AA1E58DDEFE7BCC9B3C3A04EFA49769A50F81AA8F3D152757F33E935CC6ECE9"
01541     "959B47BB90810BFEB6108C8DE4643B24B4154CB7EC3889F5FEF9045A59E82300275EB4D379"
01542     "BD85A29915796B8125351C1CA84EAEEB7607894508",  // PRN 42
01543     "C9ED53081796789261D7D209BF086C92C36B574AFC6ED55C3410E0723C799696D9E7DDDAFB"
01544     "114606EBF4256461FDC9B86BB21A134F735837E6845571F6C2712AE42FE113878AAA18C76D"
01545     "5383FBDD38E43FD1314166DC0B7CFCF3F07ACA56740A582173D2DF20F7E94F0CE29693AEAC"
01546     "6B3D1E4B22BC63395BDA6B49E30B49261FE045CAC9AC29BE5A2D91347EE4543495CCC1A38C"
01547     "AA59593BC3DA05611BDC74806284B4ED73013A415E6B36BC2CE31181971BF4C7B488E78E92"
01548     "1C6AC1070772EC36A9BE74DA8AC4FF68B6210A392866B689110D9BC2C7ABB2D82A38EB9908"
01549     "91F264DDE5FEC75C6B4E539F971AF2728343F1641F4B05EE8EF3A520DC170B3D8D562A7AB7"
01550     "2C409995034F6CE1C258F486ADC9E44FEE47D5E81BF1259C0AAD3BDA4A9E17CA9C74AEE77F"
01551     "ABA25B3481C0317199D585BC75BE5A774F8D5CFFDFA9FA652C4B14E29AF2DB5ECBE682CA72"
01552     "4BD98E05BBFC82320C8883A56F687F840319DF2D8BDB3FA8553BA3FFFAE0D4EE33E84B6222"
01553     "8F924A86C5FD5B3E12672DB6377D807356144009F741325714B5B6AF0803F3DD1AF558C0B7"
01554     "6620F77056A18896D2047C967BF69ED972BDCA50409929CC2E2DE802ED09EF87433B978A85"
01555     "0ED7710508889EEA4558097A7FD83EACB365371F80FB6EE1F7A6D258B97BF1AF9C321A629A"
01556     "5AB7559CF1B82311F816836E391BDAF741E493298B49C001783921ED1569C149C90D26CBA9"
01557     "4F8604BFFC94FE303CCEDD8812A95F7B0E71DE95869D2515E524083C7B2E06C4B5E9DA12BF"
01558     "CA74049F03A352C15DF3B79C3F2015E86FA527BEA76102DCDDC863263F0C7E43CD9D49F05D"
01559     "8F6A124354036388FAE37DEEB43909E415EF9F5877D04CB5A86AFB74FCDC990623C9D4D034"
01560     "816E39B692B68B732322695C97CA93CFF0FE7FACFC1DA29BF0D0C08D93B4485D6664E96B09"
01561     "C8B011B3107FF5232DCDD9B3B8DE7BFDB4CABF31D2E92AF686EB8C2FDFCBEB80B5F5591B3F"
01562     "7AAEEE0FC7A3E4BBDA652C8E857AEE81F499D9DD537139B7A6275C64DDBDD17BFF7594DA69"
01563     "335C01AC24270403E880987F69D4EFBC2BF2B787A631859241063E7EE106D3EFFFE60479A8"
01564     "BB5C11ABE38A76804D43A44327FB4D4F986BE3C52DF109E4D72A092F25CE7FD384B0404628"
01565     "936256877CDDF543099D5B0389A9E05CA93A2B12C3C89BD76F55084A03148E29275159DD41"
01566     "A75FDDC75994B8E8D9FB10E6E80A4E30E2CCE8F8DF3FD7A8DD567BEFF109ECFE3B8C843BAF"
01567     "FEEF5B70D5747471FB84DE2A26747DC73C9B2BE9905C53743159E489E1D0CC14E72046E4C8"
01568     "D84F893CFC7EF15CD9F52C71D91A503802FB5A6813A7BD2DB705BADB58C06E0883F0848E44"
01569     "C20A4948119058226031B5731F36B3A9177065BE9A0294F726E80B5AD2F2723B8698D831DB"
01570     "A376061FC2029979F0EA7AC4BCCD87B6A432051DDA7AB2387260082E6D3BFFD3B2ACE807FB"
01571     "6792DB21AC235C1A58D9E47D42EE055E716D389066AB0814B36291AC71B0AACB666F7CAEA3"
01572     "7B89CB424C4EC3CBC68F97AD6FCCF5F3B5094E60D68C2F1180E5CF2141EFC84DAE75094901"
01573     "38E0EE466AE0C7469F4EAB2E5ACC29EBF2BB225E595181F96700B552BC0E1171428FB2A2F0"
01574     "448463CC86DA6F7DD10FB5F789857A29FFE9535D7CA35F70640B9D4B986223454770331B5"
01575     "2E42002D123560BA7DCF9114BA9B8B23D08C3240E3EBCB7359D13767810AD3F3F66A8D698D"
01576     "17F20DA5242259090C0A9AADB9385A2897A5A1F32A7B1549A45D4EA8787523AA210BCB55AF"
01577     "6B15A1F150E79FECB104CE01DAC70125FAE9D1B8D0",  // PRN 43
01578     "334994DCE050CEED5471E648F7010ADD69AA0CD3DBB3679C9D3863F86319BEAEF1370902CF"
01579     "4978B8237638507B392AEFEF731E6FB42DFC2A23142EDCECFBA399F9C583866B1BFD7C280C"
01580     "AA43C2CCEC29C6657654C251F7BBC0F83E2C356934C825BCCCB36073A5D540AB82DB7158C7"
01581     "893AFF166B108363405B206E53B13A51BAD149AFEA2A40A37328D4CB3961C9EE2EAD29C76F"
01582     "70FB4E6409F468364DDE7A2439D24FBB05719A6A489513B14229E1949142301AB538AC775A"
01583     "F716707193959417B14A65E2D20319721FCAEE6364D04C480C07D732AE0182090C517B96DB"
```

```
01584     "8FBBF66B8B6FBBA35CDCFC87B1C8C0F2E4ADB486474F842434CE9383A57B69D4CDAE68E530"
01585     "6DFD737AB075A60073846634AA6E2A6D6F7CE56A631FE1E3DF256A1E39F0C44C3EDC3A9457"
01586     "EB4FD4477F2AE094979599B9B90EFD1BB9347DF2E654BB9687A2EFB857D9AC1306E20769FE"
01587     "C4329BD54804354E0235AE443AF3BE01E26F0D5998D7FC36DDFA5EDBD3958623B6F065627D"
01588     "6A4EC9901891E3C63522744CBC48BAC51BCC89602D66D1B3E0966398927E869E73AA8B986D"
01589     "9FBB428BB3A71AE7B273A16228349B81927B53DD9B7E96E3D078F25EE564495BFE7B7E6CC0"
01590     "42E016499952DCD37F171C709D4734CBD8D46E7989C2C497B5EB9613DC311643ED4B1B08C0"
01591     "C4C2A5854CBA30AA225C683913B5BD46E1DA483BD7EFC2C469DF017BFCB5BB07C64A68E3CE"
01592     "828A537115A07876D30999468358869AC7681380A7E462ECD497A6A6FB6AA134C15C6BDF89"
01593     "A2B8D58B875C00892C50A791D8A50514029F790F7748EB0C1CDDBB07AADEB920238F9C19BF"
01594     "A31097F9AE3B26509DE5C4040CBD82C859EE7C4D094CD7CF8F04395684BF7BAFADF75027B4"
01595     "A47A21CBCC6E3ADFA6A2AC66EA32F670F49BF245F6CDD0FECA201E3E7706F8E37C25FE87B3"
01596     "040C0D8431A00FD4B49E7A050837560D458BB5C2E93986EBED14944F9E92463C312942E6D8"
01597     "340BAF149042FF611DA8ADFF9DC86CD28185555E033F3E22856A80C482D52B2EB21E2D5162"
01598     "5FDEC689595EFC3160351B5F30E1B802D5F4F35CFA2C25D8E4101298E6B0B917B5ABEE5DAE"
01599     "3CE09DA595122AA3BB3D8BA8BF0D68EC99C2DE8B4BD7C2A0E0E32ADD32D72696C43C7CDDDF"
01600     "92DACF13D18DE4A1403D0002454E7081B92EBA207DF9411870B91CE9DCC5E26887A437726B"
01601     "6CE2B2DC51907FB108EBFD2F089F893A6D5B5141393C08F497C180AC4A95357E417D40C5E7"
01602     "FB324F0D72F417F688D45B980E215DA436CEAD3794A135A719942FD0C998F3F6F165C16612"
01603     "E2B3ECD4CFAA9A18631AD9D878D63A8E09B0F82C12EAC730E6C5A083399B7A098819F2F25F"
01604     "B2CBD1B6E65E3C30302CEC61680B699F504E5AEB959370E32B58B9FBC832F85AB0846AE4BF"
01605     "678C272CADCB5277EF13A38B1AB787F8E5344D7DDDB074A92DD68A2A2F9695FD52DD685E43"
01606     "16FA7E913DCBF34C3F0D9E9D22CA862D76DF3B5E9E3B54992A7F731C893980F151BF29EB1A"
01607     "AF677B439DB6D20AC69A5A41166799FC54FA68AFFB8D54EFC45B49E25A93B28E87E41CD7B3"
01608     "C61FF24F1D0BB74CB20D2A8D817473188FF8E56EE9C4006083BAD781C3F46BCEC777B2C72A"
01609     "9A3F551353DF186FF9D7B342D86629C4085FF2108B98D8EBC886CFE3C7A4B673D447DAC5B1"
01610     "DBF6BE6064936AAD014C0AEF0078C5D1A626830C480E8B72439C9BD0ECA0B41561E5B0D661"
01611     "8E77042FFEB6CA0F29622EB9F21B83A4560FB26BB98468F33BBFD5CB29AC4F45EE9D68A1E5"
01612     "DFA757DC218B536A9231D7FE228925F96C32E0ADC0",  // PRN 44
01613     "1B2A30E6054391704DAD57B34530294AA46DEEA044EA5E707811036742E50DECFE891FF883"
01614     "72A4AF65CAA193E1D8DDA016ECC2A2E3F6D8F2DCF9AD1449C614A70D253161E2B05AC67489"
01615     "E13345C779F3D36BFDFD7AEB2AA30604692969E0C4E375D18293D5D3CC39E31189698BCEED"
01616     "3290A2CFFEA79EDE157F2A52B4EA52FCD82DC4BBD11EABED2F311DB72EAA53A1B6811D03D3"
01617     "C2A30C76A8638E6A775EB3A51FFD4CDFFBB4B961075E6102D2E5555ACB06548D119ECB6895"
01618     "AA00394D4A98C8302E02F8D4FC5CA34B405F25D2B090932A10A4A68CA18D799C1DF1C86B1F"
01619     "88AB219D69C25CAB6AF6CA0B3A4AE390CAE3AA65D3AC95339EE734126226D1F292BA3ED968"
01620     "FD2CD835F621C3CB09E59B9DA5CD23163F573821FB64E296853A8263AC9DC896D6E355A4C4"
01621     "377BDB4E7004CF399C0891C1D23E39F03CDDD94DF7FCDA06261E923C9E0FBD8DE32B49935A"
01622     "03E66C13FE7B00E83674E64D5CBC14C0B9F62A842A17551E5E7001BBFBADAEF36228513A43"
01623     "D4864B8D2C8B272562A6B548C0B3E4610609191B807CE861538B89FA079217C91745EA61AD"
01624     "A6AA96464A135DF216FE5DA5143F55D76D44131FE6E329826F2BA062EAC329161C8D158EC7"
01625     "A3BD5FE57908C7523430ACCAEE3091FAD4B14DFFE0DFED92F014D19641EBAD22052FF8D262"
01626     "D39D6FAB21ED66A82CF8336FE8E3CB3641F15EC5F2E2AC0E99CA07223868BAB78DB3A008B2"
01627     "33FCC7966D816082C231C18165422F7123325F365D257FE69867CF0CCBBB875D5153B8C1C0"
01628     "300EBB1AFBA3371B83488143EA2562FA477D210BF68251312A7EC26BC837F9D861101B5802"
01629     "49FF24C364DA978C0F57C3A3C29F03F3696C04A3CD330247DEC04A385739D9073984786759"
01630     "2B09DF211D8B349FBAA760D64D5E00C9D596CBB8B73E7B17317AD315004B35E7F986720858"
01631     "11DCFE7C86DC0169997F2F9EC9B2B77B97FFA6CE2AA70E2A1030BD3ACC5FC3241F291D0304"
01632     "B71D727828BAD824445226CAA77E210A48EC8F79E0DBCA01505D1AB9F79642716146F7A8F9"
01633     "EA25238FCB2F9C46BDC1489D35B958C44C7928BB4503DB8DFE4C1AF5406C04575C4DAB1199"
01634     "5BAC0C7FD72B10ECAE376937B56A035DC0A4D7B2E1A180EDA4A84F97D53B57EFC5562643AA"
01635     "62C37211C2363EEBAB6A2AAC22FFD0BF623CE6867F3C50FE502FF535A68BED2FC98C50BF67"
01636     "3B2C5BE8D7D690764C256CDED17E4023A04E70387EC53947395B00E8ED6541CBF0B076AD2B"
01637     "2A26CBF220BC665598D9BC1D1EDD309E687F7D1E7D71F4B2C1B5BA020EFB1DE4248C234A68"
01638     "670916EC8A72FD1EDAF0615CF3FA768FEAA586B08990E05E24E7D8E0DB32F4370CEE72D74B"
01639     "AAB68F6E57F2B9B0F51691990DB46DFADDA7503BE7F64A13F5EB5D49788CE21BA768D02803"
01640     "7BBB06F3573CCB244CA51362374776B9EFC776AD71AD1F5635B6CB73822A112149208F10E6"
01641     "57E1297026113481BE7C862FA099F331FA5C6B1099AE77D4F6DF2C74736CCD9AE834228798"
01642     "6842063A65845ABF45BA037441D967FA102134B25BC7F1F9E8BD4E54306BF8C36DC01A500F"
01643     "4A1184D7412841329DE6B27BA4D772B1B2B68348228037A9A048C0989AB12B42FBEC45C692"
01644     "03ABF843423D51CFA2D67FA548BB39D0996D4C215157C707DC251032ABE51686BAE8F00E0C"
01645     "B4EDB664A3CABBBDEC85EA7AC7724A461F9F650F9752C1E5B8DF204C3520CEE4A51BF8D3BC"
01646     "9A0AA3AF9D31B95FACD006ECD27889F1D0D1B696DF57403EF0E6FD83164E5900691FB29B19"
01647     "BA4F64084ACFE985679876324 83F2EB2CE76E1E0B8",  // PRN 45
01648     "5513F3F1C262853F074269AF598A2E43052CA2C0141729078A17BEE4CCD35B761B1766B3C1"
01649     "18C06724B0459AF3E973110D7096044F5CAAD9A70F3BC7AABE628848D2A4891AD18CF4A3D9"
01650     "BE900A2F42F5D54AFC7FDC7DBFD3155F9D06078C89F8DB684B62E4CE0DC0AE0E4EE6F0E26A"
01651     "86CEC1DCDD96EE175FECB827BDAB3DD53D465E796FE4C49F46B0B6BEFB951BB1A679190F3F"
01652     "AEC58C00E2FCA744A97907765A3834585E4E00B952B5B7B5F10E34F43C8EA3F46C58F8E31B"
01653     "3239C0658F81FC7DDD5049FD8D67587C85C78C12CF122CCA327A7860A0E34A34DADC677F82"
01654     "014EC7217D2710112B6C0067AD402A941877970C95097A47AACB74A8DAE4C9B5F360B755EF"
01655     "C4546D7D63C580F108B197375AC08EBB1F51E8AAC68C39BEE69EB886BFC2558F0A52AE16F5"
01656     "14EC168992775060A5E474E8B2A1180C8F9FD6F3E106963DFA09D48007CC276C1464B00C86"
01657     "CEB1B7372684ED6BFED1DBFA6248F4725E7F8123904BD0552EF70A2E21B8A078CB90EBE75B"
01658     "127A1C887002D70E32CBFB6875138D8DA5B312A97AEAC2DAF9D6506FB08FAA9C334FD686F6"
01659     "CE07F4BAC70766C6FDB001BCDD43421CF70B8B33D22795303769B2C37D25752C9C4494A8E8"
01660     "A0D9A1205A03889946833C065AFECBBEB026E95987C022CFE65DED4F4CB78ABD4DB2CEFC3D"
01661     "69A4303EEA5CB2E8F39A2FCC995DC4BABA644F65411F6809891DF1C0958FB281A7852C8330"
01662     "109EE9D87362F461D3266BA8C2D3E82C9B1400220B19277649CC804A9FDE4C2730AE11CE63"
01663     "3975EB3126E8AF60803DDB5C0258F7974BA7C93499B2C2ED1E71CB64E61854D3FC18633332"
01664     "8D14A12AC06586246EC7621FBF1BCA2578F2E16A83C9A6800A378C8E280C7F3378CC710116"
01665     "81C884B83A56865F138FE35C123E28F1FD461F60836645A9F9F8CE211A8FD6B07D477674A3"
01666     "4F78E3B916A7AC73A0AD9C8323D0229706A8C2214B5ECE1301E470C37818D68092D5DA3F0F"
01667     "794B8D476878A5B9FB339999F5422EAD28A16D8F8B39CDE00D14B659AD8DC33A46F64DDB8E"
01668     "4F142FEAE8A9F55B94C40D62723CC51EF718C0DFA6D9799AA1863498173A0659F816FDEACF"
01669     "2DDCAC54C1D535E1C0FD9922B333C780C1149C4B294AFC5CD23CA8EFF423A3FEFD0EE1A5B0"
01670     "CB28964B8E80C42AB660665C4B55E06E44557689AC22196CE641644959BC7F80D1AA828DE7"
```

```
01671     "5CC21E0A8FD211BCF6185A7DA81222CAE0B632A7F98484ADD6E83A220CD49D5942B7D16309"
01672     "AB1DFB705852A22A4D51EDD96DB3E318CAD7E96D5531193DB24B50D3EBF55DF8214093ED04"
01673     "91B037E1B38FBCED71199A063ED101C227A10313051F0B4BF4142057A73C491AEDBBE916CE"
01674     "4BC05D1A44FE7A2E7DD76CBD24C13F3758BF7E3BB8139690E33631F180ABA7425DE3E512B5"
01675     "3663CF9A8D1B28C0B886C655AD3A03FA8698A5F5AB5484E3CE53CDE1F34A66FCBD734D15BE"
01676     "B727B0FDFDF68BAE40299C73A8F455ED471C82F6DCE240B3BA4EF6364A56828FE2D6BD8DFF"
01677     "03F35F5C6CBDF0C907F7473F24938016615835D801598409F1004C478BB1B2283AEAB437BE"
01678     "EA2F5FC51D8E43F78C0092ED4D0B08FFBDF1CD60454D3BFE6EA1F41EDC5EEC06357DB41F73"
01679     "CAB23556634416369365D13FC8FF9E20EDD8FDEC673F0BD6C8A1EF60E394B596B70F751EF0"
01680     "D9D7E8D573E3526D2D90325B52A08D25F18DB0E6757EBB6002251809ED47C85D4FA37C5890"
01681     "D60B40BAE3AD22258BBEC4CEAA1B6B585EEF32615F7254268DB82532BACBD9C553624AFEA6"
01682     "BE2AE9E1401F9BE05D68A881AC8D9E0CD437F60C00",  // PRN 46
01683     "7831C1BA1031B6B0E0ECE28BC7E6A8ED16E3068E8F0006DA96B8740114397B19AA19370B1E"
01684     "1F061AEE3895C30D0B7099668BE752BCD8B1CCBC497E06BB0F75AF47A36EF8925A59E07011"
01685     "5D3ED63C6D7FFE54D037B4A818703F6D78BB37725D7F0AECF6CAF120A80C938FF44BFF1654"
01686     "4C7847B1A989598F1602A220392CAAC2EBC390FD53D15CFD3372426CA41DCF816E29C731FD"
01687     "CA2E84CF90180039FB1AED0DB9630BC28D54201D5606906091A3F867D257C3A0F8193DCCAB"
01688     "2625D40DE5EDE651C20F2DBC83FD551EF46C4E724D2C516560749AA89D35C7283385E98AB3"
01689     "4BB88F8ACF50A0EDCB70888DCBBCF0B792A5D1F2F6977FA081B4EB08BBDA076A43BF1EFCF1"
01690     "C387DD5B92BEB5000B4139938CB0A3B6DFF47B729BB5826C7E188A177CF6DB1B88F051B537"
01691     "DA014B396645218DF4EFDB0DA597B3A78E9884DE4A7B3D06C259C10B78DEA1DB711CBA8FAA"
01692     "2CDF37EF0378091E408960B011F4E4FE24771D0FF3B8DD01DB0527087AC5C0F64EB693D38E"
01693     "0A128FBCB8053E811602A05F3DADF2166CE49EBA13F382E90EF4A143B2CF4FC12DD4873A40"
01694     "FF7354C6E648A56623457665402632C41942D23001C08470AAF1F3FF0A474AC6586C76915F"
01695     "C91AF074103A1CB9304485F92F9639399028EBBFDF9394B4BACD6BD3B2177368B94F639AA8"
01696     "D202D6BCA88D74AF0610C7B565067ABDE4F2BFCF8CFFAAF93274CBC913EE7DC5C77BBE978A"
01697     "B9F82B546462DB933D67E0C6748F943F39B8FD6791847C50A5A834A78C2FBE5D21448A035D"
01698     "83D45F9ECF44C21C4DE726AE8111CF277049331D78B06E9D7D3D0D8915C7C32CD6954D3704"
01699     "5E7AFF8AB496A4827574C457523AD6B85F78641BCC4C9535D29B3F8AB4D944125E6E54B3B3"
01700     "F23EBC04BAE3AB102977EF829337F51CD833B4F5D4559D79E695470F498EB4388E573D004D"
01701     "725FE41AB1B39F9526867BF7B99301E0E50AA569B3B23EF78D02FA53C4BB2A184B02977F2D"
01702     "2AB3B71DB09E3E8D9D5B13DB8AEB0A95F9296B7C49B1B3D6903BEEB1B9792B5ACC1A5D1167"
01703     "2CA5D8E3825D044DA986D01F6E1E292D74443583E49CEFBCA177BE275977BB4A4F77ECD6FE"
01704     "DC886B4B83AEF5F34AF1A78DDF6B11C201760175D57823DEE0FA5C6A9C1A91638561B857E9"
01705     "93779B6B0576994CE86E7DCCA4A89BA2ED84ADD1D8B4E11446C41DD4B1EA71352E26B2AD4B"
01706     "8A1C5C64822CE21BA4B8D7D6589FCFA906768A2BA0C5AD1758AE7B8A2BD757C8E459814CBD"
01707     "D6375DC58497279C0C327F67D14FFA5C90167941CF9AC642B1693D7DE52578AAAA56E0EEDF"
01708     "A4CC12C7469DBFAEAF5F9D3D72FF103918B6F6DF7E0DDACB208A8764B459EB01092A101D42"
01709     "E762FD5C5AB381AEA19BA6EB8CC9CE138892FEDDCBDBF818E97EA8AE9846C0CDABE9B4FD5D"
01710     "80431FE03A4CBDA566314BD1AF59AF120EEE4105E1B701A4058F1112F2A5C8D9F68EECC870"
01711     "DBE78D851F89AFACAEA3A2A068C59654EF5E5BE2F6727136908D993434E5EB1B84331119DB"
01712     "1C02547EAF7C9DD479C5B326CEA6224B16A2E108E76CDF0C54D16E472D0CBC80DAE7980713"
01713     "602FDA31A6B04E4FECCCED96AA3901D7B35D445BDBA020DDDF1BE5E9914ACE600ED24E49213"
01714     "8F82AE583A2EA454EE7548882C1A192F11345A3F5A3F2CF64B85DF6306F7FD396C4E773FA6"
01715     "6CC098EDB3CF8E5599C2EA13A6F726161BB0C0D6673F02E3DB24FF75A5698E91C1522424D4"
01716     "43165B52C90FFECFFCE6C42656138BD85409C45D8AC453D6742B42A43A7626E70C212B7318"
01717     "834B7D29CAD0C29F22E6AC3A59359B1A75F727E1DC",  // PRN 47
01718     "30B93A783620E3E094D47590AA38B10FF12C3E421887E9081F9A9C5BA1D753F341C33B7322"
01719     "187317C6610EFBD0382DBDC2D863C48F15AC3FA86B3C5A5B9EDB937AE961E8EBC3D880CC79"
01720     "141B79869F95D6E90C5C5E7D9448CDACCBDE184F6FB545324995184770CD2838B61376B148"
01721     "ADF3A42123AEFCF8890CAC7D3AA87A624DF3B883E86AA5FE6DB385778F50BD4E528C6604CF"
01722     "AE0E39BD949118EEA3972327EA355EFB98E9DB563FF72C5B860103AD262BF34BE17295E37A"
01723     "81D1C98EE6FD66909144670048A50AF09435A68FF9A70B59DF8FA7A543BBA24EF8E13AA509C"
01724     "7A311E12773405A4024A0C2D7C2802025FC48E8FC12E4B60543001971FD20F1641B968CAC2"
01725     "83BCF8140629DA13D1289B7130F3714B8318518770EE1399CECFF1D3074A65F8913596F92B"
01726     "269529B5FC63D824AC7B6390580C516BDA78BB1F158B2C69AD2FC1BC610AC24DA696F44B58"
01727     "DF813D9DA52529D325BF2B3575768DB43AC3BE5D0402C717FB587B6976E6701B0A70D04F32"
01728     "50B6C54E99983E2B1057DFD9EB7B621504A1004E7E67B33B138628EC7C43A199D00D9FAB8B"
01729     "1990A5FF29E4399051D1C62AD0D86B651E9DD94D27DD6A7B09AFBF1EC737B161A4A398F7F4"
01730     "24889C0BE6FBF8EBAE9F1A835FCF4AFD45FB36919D362CF73F0786D93552A7B4E421EFF9D6"
01731     "F98078F85098DC99A1009D6551F02B60E12BC28499A27FD7DEBE1047CD44EFC0D9081323DC"
01732     "08866DDE79F8AE34136B04787F6A5918F5AEE705CFDD5071ACF8FF54A232FE39E1D79D82EF"
01733     "FCB6CE1B68A17B4585817CC9AC33FD6CA21A00929F1EE5B9E3518B27060867624D8262EA18"
01734     "F1B01A525EB6CB337B257C8855FCAB72D7A83DE52E4F1D5E29EA697930C25109182B95A202"
01735     "E519B3757092239BFC633C4A7099290FF7B99FEBF81DF5AC41366D3DAC3384C65AD0EC84F0"
01736     "DA4A0ABF7D8E1E2F8B898FC97325878D911512F1FA5C24F009856D44123EE49C77370F1D58"
01737     "E8476542384A0E45CD852D19EE00025A8BCDCC011BD259AD7360D12B6A630BFEC8A97AC407"
01738     "81694B0D5B3D50D9E843FFAF7A023D451E282FAFFBC38BFDAFEDB34528DC792ACD499179AC"
01739     "D26AA3765523E91A137F52BC959D5969B301B1893564AB348606864916C8D79DE6FFBDB0C2"
01740     "ECE83B0A3BB846FC37C7D4D2A37B2B841CFF9820DB444FB090B430D9714FE2ED731F097FD3"
01741     "0C2ECDA7B6A27699064774149672921 9F244FBC6F1CE02F24F98D0212AD005E54F242A7785"
01742     "8622D6BA961190F97438FC4B33217A84B5EA9D3471C27D08C2D52DF19F2DE486C9A5EDBA2A"
01743     "56E1349C3BFD312B8927A9E86A97830DD0B11F505AC481FC6834F8FF7D0E67126033502EB5"
01744     "C593D15E1A56DCB03ACA1A5DBDBD6E6FA9DD3D7D152E8FC1741BD876760F99CE780EA5BDC0"
01745     "18022397587357614ED37266C69AFE4644D7D5259770B85B70E7E395D2ABCA3FAEDD5AB32C"
01746     "4B61117BBF707A58530EC435D36370D6AD8F9DA503F8994F6CD627336A731BB149D3D3E960"
01747     "75385DF8CEB3D9B48757A79C67C0E0C701A2CEFA092D5462E28D12548FA041C5407AC321FF"
01748     "049B37225742345E065720A81098B5785AF12927167ADD2F72F766292F421C290C2648E270"
01749     "C34443ABE5B95B7E635B4CBEFF74478F65C77899418C8ABEA6A4C3FE217D784823CB448A7E"
01750     "C0739C7166F4AF733BE758B6D5011EAA9B05ED995DC70E3D5A2D2298801C25049661FCF4CB"
01751     "2E9F7501C86D763CECADB0C86B348C7BF31DF7BB0CEFB4A2EE6861D3D9D441AFF4CDDB9794"
01752     "65C1ED95692C5A46F5ABEC180ACDA140AD11BCD5B8",  // PRN 48
01753     "84D5B404488AFCC35CFC6EF1CF7A848EE5EE527B7D2D2081EBD8FA8432414FCCB271A43FD6"
01754     "18EF95CAF1CEF445B50F02AFDC6B41D5C1F1FF4B6B8F2AB94F90F259383F44246B2D400C30"
01755     "12FD76E2980827C9B5476C4651293CBE9FAA77EFC53E369B94F51BE480EEAA389D88AE421E"
01756     "75EC67C76E30A6FA7EF02500372E4C9AB875EBA1C357C003938B979859B0A914D58F21B3BF"
01757     "1BE8C33A1AF079C5A0A4EA532F93A10F6D96D0A63F3C99E6F06CBC072CC5D71A517FFF40BC"
```

```
01758     "260A26E3A46688E79E733E1E2F0010846937689E2998BC769186E0B977AE0A1689245D255A"
01759     "E7D5F34715AD526E824838B62546D9569009A191BA5192535E23142236F4035C90D1405C85"
01760     "30A57B402118FDED2A5DA4B40635D2F95AC7CEAA370686F0AB23AB31DED369BEFB066909B7"
01761     "6F2C04039E49FF0CFD323628D03C26F0BD4054C127F21355C33B49F86895D5354C3BBD2C0D"
01762     "38ECD169289F7D8403AA1753B393AC06C887F98315324DBD78F9BFD1AD8880D41D4EA98956"
01763     "EC89BBDAD08F7CFEBF26988EB5AB01D4E49AA8273CAB6AFDFA1CC1C95EDF85284E6570332A"
01764     "0B8242FBA495D87A01A945E61CD36E865CD4F430F04A3B3EE74CE0486B5676F7A938B18E1D"
01765     "B2DAE2FE1C04B3D56892C7394D0B07C5A2F7698D196711DC83FFF6ABDD4E8131E1106EBA23"
01766     "245FB0E2A696E440FB6B0B1CB4DD22C9720AC76F09863711647A7E369FDE5D6911A5FCF987"
01767     "470B8C06C4EF7063B08A8320B4371866789147E9D7C5D477A662A501627F91E95C6E2CB681"
01768     "4651E9DFACD67CA0B1FE62F44081E3BDDB964C1E5ADDC093AD185F0203B4C0F7E3E96811F1"
01769     "4C79BC54F2919FCE5B653F6845CF1AC34DA9E4CF52EBA626B739246E25204F9EF2CCF4F5AD"
01770     "BFC53CD8F629FDB82186CB52BEB7136F105D3CFF9CCCC2610BF2C8F943DF5BD03877ED7AFA"
01771     "25CA81C4063E1B97452A937FCF77F1AA068FE2A26F6ECDF662ACDF18391AF7D627C2AA07CF"
01772     "5597B5F33AA0ED4DAC98AA73C8321EBF6918EE568ADEB1A823527AD75CD5BF7ED3EB5633A4"
01773     "EDC8C59DB0A91DA79C248D6894CAF9CB411F302635BEEF1A1C7916C7545B8BE7C577692245"
01774     "E285D32B4FD18E1478F88F8373E22A5CEC6D22EACAD2A41612F05AB2C54A1C03C0512359B0"
01775     "F9C5F91615EF2EA80DE97230480F489FEDA38D52FC84E2F1258CB20FF0E850639B31C2958B"
01776     "A6C064D0D0F4AAEFF313E43C65EFFFBB47CF09F2A122D15F76E8B704DA8F2B8C71449AA497"
01777     "74EBB7B2DC97CF6004D2FF6D37B9C689261189B85CFD50C20C961F22A644F5149737742655"
01778     "6B956DB8C899045B175A8B9CB22EFDE535CF487E9D958281E0467539853054FC3475D2E142"
01779     "D58F9201C1DC6B2C1A12B2878F9366E2216405069B7D03852AB37EE83112EF2EE172F4E531"
01780     "7A41653A656A06F6633AF59062A47263A9977A50B9115456A58F9C3424FE7E0CC57DA70FEE"
01781     "0411C1479B4CC2E6E720B01432CFB3C503983B37FFBE38C92E50D6C5795443F73244F6D28A"
01782     "E6270E27D46A25621F86A2BC4259DE6C04CCAA657FE656C640833B290F00A1660329A3F09F"
01783     "FE60152A23925F4D8B0933AC016B5802CA863F66F8CEB5C8383B1180A515AE6C51B5F56597"
01784     "C3A004F8D25F8A6235C97B4DE38468B63E3B859A487AFC4320598EB6D143E5C914409D25F6"
01785     "DFF6E957D575C71D3ECBC6743CCCC03E5F34791744637A994F3ADF86966B4FE911C06F7BA1"
01786     "A6C20A2971D82B64E49721A530D9D5A2319D5BE0F47B8119C5A835CABEC735A935E23CB439"
01787     "970437C6F4CCDE2525EF0D7B1555CBE70280D5E760",  // PRN 49
01788     "A5029C9EB4623226D321FF78D3C4EAB1F672A8B2B24A09CAEF21F561A851323C05A3C5E136"
01789     "A2DA7104ED19FBFFCAC3FC49B6D598F3060E93552EA6C700B837F7CE04721919B9C96A57B4"
01790     "2AA1D832307C7A847091848066CB84947BE5F6B54654479E39F654819D3EF7AF4939FB9F4C"
01791     "9B20CA7F83DA0FDA2F171FDB72455B7ED0D43206992520CDD86B29C48BCC687573AAC4217D"
01792     "0B7DCF852811ADC3ECADDB2B34B4572675CCAEFDAAA01F83561E6E240878F229698185A80E"
01793     "6FECEE89455A72A377C24CE3FECAEC2A34B9CA98D288596D1C769CDEAB06871316CA7D1DDC"
01794     "5862E6282DECE33F362C64A73E57AB266715068932ECC31E62AD2ECFF7C6FBFE213384DA08"
01795     "6ADF49E30F432EE2C715D9AC4DEF53A7B09B0D722CAC560FE8CF0059B80428458282F7E817"
01796     "17647D72E321A3E4BEF16FAEF76009BD98B8D9822B771EB62F1D0748E462FB7F3BEE9B12AA"
01797     "86D9629085AEDF8E43E1252EE59970CFF66A6F865C7651EF83F8FE10EB0E2615BF8F5C7F12"
01798     "FC601B0CF795C0D8B7057F54408BEFF86747C0F6F23EF212A9086EADD464A25341AB71FBA4"
01799     "ADED8F599C38FC15E790A5B86E64977C5AC718DD0B47C1A476AC9D7369396144F6288E84F7"
01800     "FFBDE02EC00EEAE8ED415C84648364ECBEC42164514D3E26BFD3187E0641C216FFC57E00DD"
01801     "752CDA581686916221DCD1AF07582391C5FBEF047FD1B7B956B458DE925C02A756FE197233"
01802     "E0304D0E034FF9A176B5B3F5FB683AB41D2691E13F97B3F4EB332388513311197C49C602332"
01803     "32DA0E2610430461876FF6F77FF3CCAF1BB2424B8B347588667B48480476D40BD9E487468C"
01804     "D5AFEB597C750A5E665B4E7C4C169ED08ADFC731FEA928052C4FB85B3064EC07B0CB988E32"
01805     "4893B3F084291D964403F0350B7E1B06DFB73362C38318B762A972972BFB76CC5C08B5D47D"
01806     "BA0F3A2473D7749DE9F49F50C4C1620A9EE9FE56296124D72906497411DB87D4D8EC4E1F79"
01807     "BEF27232008A2299F5317FC1A6F455F1B827F1712BC01814F0B9D0CC162B25B804278B9C7B"
01808     "C5FC5616B317F2050234A7AF92FE35A59E22C959C7163DFA5F142022BE5CC4D5EF16D21821"
01809     "6C57C2E29DA926436C00DCB82E68E16CA5A07158D8B8864D38A765D14E82175114A28CD97D"
01810     "11D564C8C7B87411589A4FBD49F9900D08939B7A73B5E6466B6F607F8AD22120A559A02BFC"
01811     "EF6456E7AECE8C9B7C9D2B322D2197124C05363B2CBFA58B74CD88877F22A5E5C202FC2C33"
01812     "531125F1518F4F0F38FA788E5E6B3307A75EC73E545391CEA200243DD6D25A5B8654A00B82"
01813     "BA57437BF0ACCB0ED37ED2FED221E54EC12B93AFA6E39392235960075F4C47340355D7222A8"
01814     "234A1F65EDDA42FFF5D19F7FACBF09AA77E7962F4CFBF61A0F26FB18E31A504B3717140488"
01815     "74BEB286AFF71B43E4739A17E8AC25FA77121ABBE6E99754AF42F1D0021EA1E3FF088D0734"
01816     "BB191F91A520C96E22B4A28F9A2BD7DF81E8079EE5D0DDCBD517046F12098FAF6920E0EBA1"
01817     "0DE8CFB391C63C60D62C1F4BB26BF8B6E421A830575731F67D306CEB5D6FF04637144790EC"
01818     "4AA2F435906320114CB81EB40C22B271FBB065474687AA5880F1DBAAA1744AB3E9B831A932"
01819     "A9208BEA9F5D526C52F5FDA56320E123CFB553E2B71A595DDED2ECBBD6E890B0421D765D2E"
01820     "9FD0D3995DF2A9523A65FE2040710DF16F2A83F510DCA08493DC138541E5681B51EE87D84C"
01821     "9AC11612EB5C06F5A63E22BD6275E35216766D79B215DBD087E9CADA0CEB09BFE435DF9B78"
01822     "09A76DE323B373682B8C58CB4F08D9C708EB050DEC",  // PRN 50
01823 };
01824
01825 // Galileo E5a-Q primary codes
01826 constexpr size_t GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH = 2558;
01827 constexpr char GALILEO_E5A_Q_PRIMARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][2559] = {
01828     "515537AD5E5F4216C16046FB0AC50DCDBE5CEE7E3CBB51B6ABB4E87A407B90E0EFD49DE1DE"
01829     "5ED29184E7FF0DC31F75FBB94F46FF6586B36C7771E5A68D060A965ACCF8D640C6B6E4530F"
01830     "DF19DD2491BCAB69ACBCFD3EC7281CCC31253A471B652E21C4CB0B43613EC542266460F0A6"
01831     "199B436BEFD95572DEBEE920A915FD854D17FFD0DF8C74E23B21B28493A0927709709B07C6"
01832     "5878C43B69DC501E9D0AA21061ECF173876CAE708C764435832D9D6FCFE62DDF2543016D63"
01833     "25A56D9BF1007886E62E8A832BC32063CB0717D723C5E8C5F0C0EB3960577D364C93060B64"
01834     "EE04A539B7601CC3113E0AEC53CF21AFAD0154DC5CCECF038474E0F4004A65B1EE2801F819"
01835     "68B88C3D35E87CBB126C02D770CC3D32A552883D351DEF47847391484F80646728221F9939"
01836     "21BFC14126EE3D9527DE607152724C6D2DD305D3FEA0AAAEDF6509A2FE3248494A54FDA8E3"
01837     "CE7E6BBCE234E4686BA5A19724BA2CB78CFE71A6AF45532EFB286C5BB47BC3C1EEF4E4A8C7"
01838     "57786AE974F30A86CD60EBCBFDF5502AA8F643819CBA4301E731ADBA1345B61C0B444FE7B8"
01839     "17EA86F8DD749C451AE7D24A68D914F26C918238953E8AE61CC8553213DD6856C7863F9F6B"
01840     "AB1B4C84B225911E7B92BFFC12AC211B2B2CD905877FE976E07057963D47C437FE47D89648"
01841     "053F81AC39E8FD2F3A726866F6693E503CB6F0C3F0AA9B3EE2EA3BCDB16D726E1C6D8B073A"
01842     "A15F64EB68D53B1F8CDAC19C7AC33361226E81F1C793BF188755A3FE1BAC38B91ABBD4F077"
01843     "F7A28983EAFADC346CB941D49492625893453B364D07FE06FE42B160C16FE0462AB6366FFD"
01844     "EE54DC9CE4DCCA21E4E4AE5E92C872D1E4EC6FF6D3063C98A5AA5EE72481A0BDF15152E2A5"
```

```
01845     "425AB722101474D0E1EC8401273EA1BE1DAF7403190A94305BD1C7DFBE1F35F65D5CB97E82"
01846     "B7A297047507FFA0012FB73360FB8719C174E78A989A96E60A9184B3F3A8188DE100AB3619"
01847     "21D38E8142859C8F0F7D441DB1B2E9687BBD1086643987C83DEE0BE8CED4C83BCC82B62B45"
01848     "311CE4F13ABC55BF5EB1ECDF15F5A07F8B2C42F07FACE0E299E87727E2D534FEBF7B9C3894"
01849     "CC3E2E4127A294B9FA2A671273B174DBB81D247CD2846116500A072DC3962C65FFECD0C0B4"
01850     "6DC2AF52882058259C26FDE50BEB319AEECFA1FABA34C069680B9EBAA9D96EEBD7EA30E748"
01851     "213E1283396A2AFC63527624641D4E1F1022A973B1898BD4CEF4D712B49371A51D60E08F42"
01852     "ED1EA90AC49EEFBCC53E7F9E899DD1AA4056F11462DF1A4C81620A73C831CEB897430A2225"
01853     "2B901EC3D6F3DF58EF26422F796EA31AA4E0E9CE5B4A9C312A22305E298FEB3B3628283D40"
01854     "5EDF726937327D90C542434BA3B60684584A9DB244839D2ACBCD7EF147A541E35687B5B8F5"
01855     "F07764973112D20D1ED75DC31F6A938542B42EFAAEE0F11B0583AA4925C3132356200E8D6B"
01856     "DB3127B975F4115A7A8A1C471836E3C5450B501A24D4A1308BB319AA827222B550F253F64B"
01857     "6F7D2322C6A2D3012FEC265A66A60102A3340CBDAB900DFDB36693D41DAD8DDB8875F8C3BE"
01858     "76AD5355DD81D67AAEBFFFE9458E522BE0312E60F63DD92F25C0D7CF82F223AEC0BD745675"
01859     "2CBD5151FEB5368F8857EAFAA90E8C7499B75D46EC4CA20BA8A24C90C016B5BD2CD7864828"
01860     "C6140E98EDB9509AD1194F56D49675D077DE92CD481B469E3A37F7DF0D5392DA4CE4CB2825"
01861     "30F1C73482CC0926B877B00B0CE49FAD21E4C26194C7E950E0078F3854EF88755E08E93801"
01862     "65C584A3DBF1ECEF6A31B224FC321326B93797BFE8",  // PRN 01
01863     "D67539AFB80711A0BA3CD67D963BAD346BA813D35A2EEA104D36AAAB863C656A07AD61BA60"
01864     "598C07744D32ED01EFCE928346C09EAAC2D392E5655F0FEBC486815AE30A38014DD8520F73"
01865     "CCBB71D9D42636328A50998A2A3BED3E4B34D0DCD65B94807064E2EF0C420898DB96E3B99E"
01866     "A9A0AD91C63857DEAEDDA5E644E62212B23D72FEDBBAA78C6581C677B10689C4AF387626DA"
01867     "C55F4EBE1893D52D28D20EA365702448A64A0C553ED337C3BB911DDAE2A91727299D8064BE"
01868     "C8801830645748B5E3631E70A0590210143F4079C572BD5E2F7634C2D53B1FB1DABE79C4847"
01869     "99E7075EFF98F033F5B2EC66C373825335D883911CEC9CFBBE2E38129B7E03D9646A7E513D"
01870     "5069043BC62AF4C524F12D8F98D8C9DAC5D8642DFFF48CF6737AFBBCBE965925F55F03BBD5"
01871     "123C9DB47AAC780301DE91FBED3C01D03E6464C2C2915BFA187A4BD93E20C24574FB91F035"
01872     "8CBE0921DCC8D6B7E9976763A1D2158511861EFF5D1C0B71F608E7ADE91D9DCFF5640B55BC"
01873     "9BD1BB322C879C7EB5C06EB2601D06241D09CFB1BC1695DFA55FA044E0E2E4BC86EFBF6A55"
01874     "740C4640512DDD6CA069940BC0FE1738FD376C68BA8AF7CCBA7D89F7966B29355538836372"
01875     "EF418D149EBF3AC104919D91BCA2F13E79A7CF7684A4DC0AC556A2843E041A71F97C94B859"
01876     "FE009659F593EFFEBAA6F6C1C57A5BF22752613AFB26379C42AC25804AEDAE22D63B230FCB"
01877     "858F496B8EA6F37104D0890525DBDE06AD988BA0287B0938572F14A98EC9E60E973FD693DB"
01878     "1F2AFF671AB03FB12B729D0867938DF6B60EC69790C992C6C33A531FE56D0ECC1465F65E3E"
01879     "57FC9E45F0F65A1061CE6D3190B6C1B8708A8A5A47222369AE889D26499CDE8F0548B8D707"
01880     "1F2D4DF6C0F2418BE449552327981CB0B54F792F29A71E30DE257CE1B3A7553A22275E4C8B"
01881     "9FAC3B8D4E9912BB22B0A899E7337513C7ACBEDA15FAD3D6919FDDD941CA659D78B74FD39E"
01882     "2E2F622691B89CA82FFE602DB2578A20D4665184456F32DF4DD6CBC412EC7C6914CA427CEE"
01883     "02F6D9810AAE1406DD68ED3869BD8E3947A2B3A803E875FF82005D853E3F43A6BF93603074"
01884     "4C34C8B71B7722BE0AD3F475E531C9249A42671D5F3A5C77C4C28DE29AE953EBFB572578B1"
01885     "7B636F8365FC755C22871E7D53A1F1561C92909305C9FD36AAF79E8844B63370B800B25CC1"
01886     "355211D9919B830A988926829F808DD2C66400279E6AC14F8EECBBE8B6E9ABFA3BDBB38A49"
01887     "535F64F719EF48C5FE6B2738DC6F71AACD70274FD40A29BFCAA594AC3E7D0C3C522E406BA6"
01888     "392444C9F362339E8FF34BE330911DC7EB11A47FF3A62A46CDE961A40CD5B24020909E5B03"
01889     "4F45FE96CB156FFE8E2FDBC12A7C12D60D24BFBE596544E4F03AF26F086A5A667496B7DF30"
01890     "2E4DCFC568C7ABD665EA7EAD8A7F5A000DAD9F43E68C4D8A14742E050769B3CB270E3856D7"
01891     "E8F4E827046D3E55A52F0E02C883881914DE87AE3C24D93E61A94919B40398D3EABB1B5142"
01892     "431AB919208A9785962D05061EDC951C83C73FBF6AE8DD6FF839E631C9FCF6635FA053DBCF"
01893     "932E359F83FBB3EE310281569741E3A8975FAAD1E573E0EC3207F6DBCDB8CC90DCA1CE517C"
01894     "8DFC1D31AE4841F87A5157792738DAEE7C29240DAE26C6C3E9D8A899D2271B0C374BD2EE84"
01895     "6C6625E31B2F8379A96F601323039D281608A01A1844E5D1D20CF1D92C52107CDA71BB3B7E"
01896     "C4DEA958CFD7A79F71868116CD1DD2B2E66BB94D1373B733F324BE489DA51BE72B01A85728"
01897     "80AE1E61650B839D03B0192D290D5B36A9CF93B304",  // PRN 02
01898     "58B2E58A13D4D5F84F37A389E6B01DFA66DBC6A25338B2884EF08056BF0A9124DA29254AA7"
01899     "9CDF1B0944DDED4BC7FE683EF7A3C7A1C359E61E959471E30F9D534F43EEF274AC6535C616"
01900     "ADC7455BFBFF43ABF268F7C995CE020CEC73BDBD04007562F2710498AD1A324F25A6233B2D"
01901     "F2D9A429F1C39943E45F934986D1979D293F90CFBEBE01665C98C0D72D09A382348136980B"
01902     "31F5D4696B85C3F42D5C445996804159CD4C4CE7547C4A3FB718AB62D9E9826D05C44C6BD2"
01903     "1D708CABB01F46514F29FBE7352866EDBFABD0ADBB6093E1A519D0E11E27737239A0786675"
01904     "2B6D864686AF1E308481C53C635046C0756E8008D3CD26DB970C5D6EF8CB9DE6BC4339EE9F"
01905     "88EA45D11B74A1525E2F1B9F91567A78E403F7ECD47A99E95C098870B928E1B0DAA984CCEC"
01906     "A13C6857F350E808BBF66AB4EDFB0F369F6F009268D19DEEB377F7D0C4ED6414EF6D23A0A5"
01907     "F37A25AAD3CD62C2115CB70409049CF0CE6D4DBCC5C96767614C6FAF73E9A76C7550BB970A"
01908     "B3D3ACE0C07FAA9F36F1935DFCD5228427BBEB5FC7A2F951D9210A5BC32ACCA7E78AFD5FD4"
01909     "60CE2A79348714E5542BF2D453680B070E85F3244E8A6B4EB6ED49F803E60B8A383C8BE628"
01910     "3B1C4DF79E0C4A23D5DAFCB26DFC5DF1D1520FB6CDA23A05A1513F751EBD0143C2B9F5494B"
01911     "AA74F9F95F189C5767C6F5623559D9F20CC9B9C9ADFE285FA4E1BBDA481A52ABAB36A53933"
01912     "70091A49A59B968CFAE89BEAF33BCA5C2A9BCC1758CEEFC2D0A175C8A5CAD5C7E3FA706C2F"
01913     "EA55F4971A7B8A0C5D1F22A26D7DA9A70603AC34566E4929AE78C8F1926239950EC0A6E1B3"
01914     "7D919E24431E53FB08B2F7DCDCA2EF4177BD7D2F81BFC784FBEA0471831CDCBB9E11D60C53"
01915     "D0E062E8BB8D98B8EE5A40960EBE10FA642751B96E1407B38A024DC64D5C2005E71198EF39"
01916     "4673A4A0097187D2475CF27B3EAA7955AD9F412A89B36AFB27FDF7FE699B2CC8C03F7FA405"
01917     "49178DF8A3C1A39441EAE869BBC89D583ECB18E310885F33B95B719045CA6955720CA75D5C"
01918     "F0B29E5F1B9E9EAC5EC92980B2D37EF6509CCA6E67544A8116494AC7EB9032E1E8CDB053B5"
01919     "AC0F60DE59BBFB78E3491D50C7ADED95042A8885FE284E90C04FFC0370B92B68379B2E7D52"
01920     "392A42FEB26B2419CC64412BFFC036C01109B9EAEC5FAF485F0D61C37A703EAD02127611F9"
01921     "D25E4EC515CABE21247156F0779CAB57C35581646BB71E87941AB5D06FB3C06C423E9E83E0"
01922     "C07E611C89CB000344AF49498EF5C30305DDC8958F45A45E1218837269DEEDBB0AE51D2AD8"
01923     "B41BE425EB7AB798856911B6F0DC721756A8154B9D1CCD61092D16A3E9E1CD5E1D7C2C45D2"
01924     "36C9B21A0ED64AD4C0660FE42FC0B543A34FB545B0D88D42F696D53BF54A1A259486846E81"
01925     "B44C16EF146310AEFB3933BA9F9D4497E74AFE1449D69105B9F295DB24B4563C3FC0166CB7"
01926     "0F104B3360BC0998EFEBABC6276F570F7679865036DB59B51AD2F91C0CCD3BCF5C07492765"
01927     "D480494AB5B87E83FD04B75A35F88656D3329AA5FD550DBF493B0CD2C19DEBE4358661C72A"
01928     "71FB17C812F75BEB61302AAAA1F757288C1D8626461F4162A2EC6296B3CCABE21B4F0695F2"
01929     "D2CB02B86035251FC9C5F2FB5BBF28237417DC56471629B3B4C25AFEDC4C9CD108ADAD9DB1"
01930     "330AB680E4998C5B3D99BCE8F08156B630A63EF36A51C813CE22A6B178683EDA389ACF5865"
01931     "3582BC2AE8170ADEC4BE03C04CBA603C0E4572A74BD9599A8442E894EB504F7703F2353A8A"
```

```
01932     "3525EC5535E750303AC07E08D952755F97AF236014",  // PRN 03
01933     "3059141DB31B84555DDC1C5F40372C50BEF6E82B433D87603BB7803FDE8FFD74DE06809357"
01934     "F11C1D6ABF02D4DB8EBE1D5ABE7F30C91A451EE3C1AA36735FAB9185C785F05D28BD470D43"
01935     "3ED4ADDF36406487CA1710ACAB57EA0DD577A49C841F5E78B70FDE90BC1087E79E27E53A4B"
01936     "131C06BF33843BFEBDC0C2A207510590E52158C0855C40146AFCB75DEEEB872ADC5E77CD1D"
01937     "D6A66EA96CC223B3295C136E2766ECE9F2CB807C68C84FD04E0A5179D58473860962083DA8"
01938     "D68B70DB95C0EB91679CF5E1BA468786725EDFB179A2C5E7C28B396A53D89A98C7B79867E2"
01939     "40176A9DF9FB0BECCC22A526B617CB8FC2B05A8DE411C8C5F9BE682459B487B26643FA894F"
01940     "9B3C97282EAE397A03DD15F07FBDF7432DD95D29E6D50CE95860BEC2649E482F40F7B83A13"
01941     "135A7C71D83ABA44A363C864BCD78050AC69A8DC0A9E601859087DBD49AA01A75053792F74"
01942     "D9A8EE726608D1F009D06363F6E5A463BC3621789187 37C5F3E71EBF130691A2048F07F3F8"
01943     "CDAB70C9D0F7B9A1180046076E1B894AFC620244B30A571DF359C7D60969A9436F05417DB5"
01944     "759B2D3ABDF6A238B5B5DCA8A0E4C27F7078DBB6600FC0105463E745A321884A4F6C5963B1"
01945     "88919EC24C460F1B9CB9C063B2CD1FD5F49C06AC2E61EEA55056A6ED08B6C7C750E30EE665"
01946     "08FF243FE1B5C494E8EAEE5467AA0ADEA834523C536855D3BF5ACFFFC0802365B3889C3191"
01947     "0A5E63B9457E46B4B7E29189F1BC21E7950BF9FB44BF5D76078D8602FFD986E1A1F1A74C67"
01948     "7BD8811FC6992F39384CB3F9B2C91F381DC01D4E79D2E66E6D76796728E0C48F71817ED182"
01949     "759D8B10F7B5D361D7138CAFBECA3234B9F2F4A76FB00DEC05D67FA8E7217E488688CF87D5"
01950     "9AB4FE7B13C793C1A1711C401B49C381F76B5CE120BDFDF30A858DE62A3F4331E4A3CC1670"
01951     "30829906935FAC167A7327B5FC0527E014801A27FE0F2D40EC13C66235EFED070979576DA7"
01952     "268D1871E8487C4DFFC62D27B83488A73FDDEE1470832EAEFF621CE8E126267A47EC007739"
01953     "22BE480C7F26C8E4785D886BF8E6518DF13DA86BB2925A432D479E8B8FA9F92442891F71AC"
01954     "7C05C0BB0E4396158A492BD49E42AB4C4737A952D6B2B2780EAA73DEE13037D659A43DE1A8"
01955     "F0E525D7F96FDD1629FE51560F77E4A1CFE903DF0891B39F50B71E0F84965413DB34B1EE95"
01956     "CC1B6728AC3D1924B34D1BAD4A4E4F7D98D1B938C567B6F3C95770068775F78EFA147ACC61"
01957     "9AFD0B6D8C66034C21B2991419217256D88F265922FD49B4BB38D2955305A8D01D01758B01"
01958     "134196C525315726D9BB6D77917A57F2C46A05DE91A3FF80DD2B5AAAD899D679EB55341FAC"
01959     "6D5C2E9C6B703806E342B2C11CF84B394BFDFCA378B26C7E0CD23C8A87B7CBEF6828D9E023"
01960     "B2B7060A0FEEE0D29CE59B379A02AAB7DFCC41A35CB9A94408A321BF78A54D2D4DB0E17010"
01961     "1798F6622D8087A2022A993E85DEB5CBF0F4DC2A047877EEC9DD145471D57015DD59A37BA6"
01962     "0C9BE39A4AFB92289562601BCE8102C83292E2ED842470D715D34F11981BCCB81ACB443FA7"
01963     "32F792C5F11B7DA5D1BCDD84EB80820EB2BA3813F5E6EC5300E622DD81B1FDE8C786387864"
01964     "D78246A432CE245E0883745D9A1DE9D68253CCBFE7C00EA908E452DD3B9669F1E6812E40D9"
01965     "E2C423180BCA98F4591307CF8B8E98B7F828989CA7F3F0BEEA4D408898364C6DC160C94B89"
01966     "D879B1D07286EAA3FFBC6CD7FFE5700304AF3FC049098590DD7F6D4770ED66FF60DB2909C2"
01967     "DC6DC67ACFDB7CDEEF1B714BFE4CA04F2D5172137C",  // PRN 04
01968     "4427106DF31EF72E43B6C75CB84BE5E375B6B6D4D21226D0FD689E8F14EBB81372B93F5455"
01969     "ACE6C168345234B378500BE6612FC10536607E85B884AEC780550F2A26FBF0445E0AE9C0F4"
01970     "CB95F01BBC76652A9E6FC457D6BA425EA8B9457A6F3C0AD2A0FE7BDD1C1CB19A9F1A8815D8"
01971     "4EB5843AECA4DD005378111E9AFBA9C3CFD808C0C3CF2B39B1962EA44B08484527781 68F7F"
01972     "60034A68E5980EB63B94E50D170C680FD6345F12CF30E9089A7C0A422661DB7B1BCF248023"
01973     "8AD043FA1D0DADEE998F7CD69FF04BC336BBD5985315D4E1CC78558D4E235CD7E05015A573"
01974     "4896597F16E65E45E0C819CB0B895A38883ABE4F439D10195D88366CD14F45947716AAFD77"
01975     "0C27FAF6374CDEB53B118277906F7E61C583E8C7CE3FE733541882C511936076DBCF65094E"
01976     "37FE2FF54E0977595077EA26E03711FE9054E2D31AE7F99939B415F46C05BB55BA7BC4F31F"
01977     "337D624A94C9629526ED51A9B93B1B3C5551F9B92A30759D7A40025E98E50128232A706CF6"
01978     "C3F6734D00571F91307EFDBC9718D78B3B792909C1C56BB8526CF5B229D4DB47E61D962538"
01979     "BBB17E6CE8CFB7854EFEB5591CDFD5DD8BB22FCA50E4D97BFC4C36E2573C0B495FEC9830FA"
01980     "C229976FB651B82DBA34173F5635875B1460A023CDA65333138C89DC81275F7F3446472F5F"
01981     "8C5F93AB130009F5B797D9ED536ADF5A42ABF6345B1457D5C96CFCBFFBA56CF124E78FEBF8"
01982     "A8BCDFFA97831A589D177AD260499130EECF8247FA5C44B3C0C19AFE6BBB0A7B87080E14F3"
01983     "C3457C8424557BD9078DEC09A3A6704E1D77049A1A4431E423B28B9476EAB6292CA1B3B4B4"
01984     "3E18EF929EFFCDD574417B15A4DC1269E593F884DF5D492A464AC3F2436B8C691B88658180"
01985     "E53228AD83C4FB843F8571D42A9BDEF37EC4926DCEE5002A12144600C9CE7AFE483F3A64E4"
01986     "80CD09F8CAFEABC9710E01A8120E84BE5861D75705A8D18F02756D8FA78392F1C89D05F37C"
01987     "B9F768289B8CA99E22E86F8200770D6DD2D76951A45F73C0730162C3638862825244360D029"
01988     "786E7CFA2EFBCD3D05169D362DC8BD472AB376C12097443080F3C83AF1EF7339D8FCB85FA0"
01989     "98BBCB5EC3645A03902B758DB263A21C8CC4874F96D672EB192BC1A183A1B271BC1D6429CE"
01990     "AA8208F5CB240F451A71B6F884537ACD49471010ACDF28887F6A02A04E738CF9FB5B00E0C4"
01991     "74C460EED1F791F4EAB3F9AEC225B40CCE223D23D07CF0B60846C111A07A86448B9CF11975"
01992     "4935E8C350752CC60AACB55C71F4D5F5F2BDC3E412039D336ABC6F3D46EB0D1C2080CF260E"
01993     "E77F5C73A35C23396E2324998B7DD375537DD35B5F27C3B545492F682C4B89F13F65FC4F72"
01994     "539E4A7763BF4AE2443FE0A2F683A86274672FB9581668CD2B075C274B242536087C4DA1E6"
01995     "9D0712161B868ABE86C98468FE43A42DD2AE30089ED92B7A750C73B19A6D70FBC0AC08299E"
01996     "E2A9BC94652B35D5B2D679220D34D3372AD756F842861470CC3A3F2BEE75112B138613A40F"
01997     "A585E449FADD6353A09363EB6025D4B81F9F224817FC2DEEAE01D797AA8A0F8C945FC69D58"
01998     "91355C28ACAB8997A0518C8F44659538191139 14A7CB472EABDD1A8D943A82FA41315D006E"
01999     "54AEA4CAB21601A2ECF6CA0521579A6A92DFB6B77BEC6D5A05B0A72F16B0328C860FAB66A1"
02000     "D6113B7098FE50F3B09C2306D0B9D00987B351CB2BE099A7AA7EB7AA691A4E2D243DD7D0D8"
02001     "64CF80A4D2FACBEC3C1CB222883CED667477D189100A5677CC3FF0DF31D1119D5F31ED2120"
02002     "E5DAAFB99E8F36E77B3B078C47B3D527DFD6521D7C",  // PRN 05
02003     "593CF84751C21D591BB62FA0B0AB65E993408A016415D6296E0F080FA9149A4C31D8B2EA68"
02004     "CD7209E2FB0B4BCCF654B013D92AD7A4F6B1935995FD106663CD5760E63702196F63563DC9"
02005     "94570C1B8E9B0A7705002ED56D335632ABE3ED8BC6CAF109F0588040DFD167DC364EA7F692"
02006     "D5F59C5616A6A7DA0C5EAA4D9FF017820225D5B164B6106CE9307B56EB08C563F123A05A4C"
02007     "93488DF63A6E4274BF5188475E8F7ABB8C3E7A8323F689DD93B043BD98BC948A567E8C6C95"
02008     "FD3788ABE728F7B7E299C460F9B35A59BA4429BF417B9F8C54F2DC475A7109B9C25C0843AA"
02009     "86ADAEF388A9915E65EA08C96C4C84022368E79A3A466B247EE6A37918CA0A8DC6AFE291CB"
02010     "9D360D727B6E7415D360AC414BD34DA40A1D995EE1303896465D52707A4F31A30C7B7DF936"
02011     "391435A5FD3F06E336CDA36E473E3D990A6F04FB5E6340581AE86D8EF81B9BA68979058BCF"
02012     "A1363D3F711D33EE9E3A8402A14AFEBF339BA7C34FA4EA2004FFD09129667646A79B6322CF"
02013     "00E1B46C1418557E0E62E106901176CA771979F4B6B299BEEFEF7B847F4E3ED99F56D44F5C"
02014     "73F00488E1DD862BFFE5290DBA737FAFA0C9D95C8307A10DDD309C177BFC46A9F6D3BC86F59"
02015     "8DC1AC69B070B9AF1CA2824F456E73747A0CED38631F9AC222914825CC3898F04F6AC09B01"
02016     "EF4CB09A63C9436625E2D0AC3C433A31D86948CE34663E5462652980B3B57C0920EB63879E"
02017     "28CA9565BEADC47F29C43F1718CE34CA5717D7AC2CEB6758A16F6D769CC277BDB483808CCA"
02018     "3A6DB99D1018888BB6A49D325891B5D6B4B77B18A5A4133AFD5E8AFA4542710483B52E3D51"
```

```
02019     "D8A4FB77D2458C4FBE2DEC1F6DD8C0FDD8FABE2130297D0606F07B305B2CEDB39204E2B8F8"
02020     "507ABD49C0FEFAD0332F8ED98736FEF5AE4FAF8515CD638E28F1555F54D7A64FB38D0B4773"
02021     "4D0AEB8B55A7922594276133733F237FC57DFC1AD49930D9844C9F44C3D6265BE3102D93E2B"
02022     "62D3B0D7760F613B1F5F5176E0C5EAE47970D30D211403477894B8D66D03CB3AE8992E16D5"
02023     "2F6B2073DED09D504B36289644241EFD21018C570A2CEE6D59667F274CBA5733E41BE3370D"
02024     "FA47464AF850B14DCC32BCF1A5A8385314E8B38B2A642AA90B8A5647DF9B32B83D3B4B2AAA"
02025     "54472BFE84EF7DF68ABD0236DE171E5AEDE1770F63823EC143B30FFC69ECC464CD755F5AFD"
02026     "2B51AB5F05B74E96F6108810CDA94F99DF828EDE147BE061FEA46CF6A84CAC720717733A17"
02027     "677BF620C4963DD1A08512CDB6A96DD3ADA995D84A934C9825FA3B588D916CE1898F6C2F75"
02028     "B75481383B27204C343CCCE41E10DFAFBB42BC6BCCFBF7BA4C460147814014F9470ECC8B99"
02029     "D677EEC872172B72E5839D263F55B9FC91AE6143FCD1BC3AEDB12312ED10647E011A020B27"
02030     "C69E35CBA94623960543DB08D67B8CFD06760450A067E3F868386975793AF43AC4D0F3773A"
02031     "FF6C2CEFC8310C0701CB934098D842604E38370182BF05A401C38B6CFBD19CD7D4BA186A59"
02032     "5ABAA48B8740DC321FC031D76F82B4F5FD6949AA101D70A702E18D8C2BC1283C2E2CB13869"
02033     "9D507FAEC5B6A092E5AECB0AD9BF2E59F175DACE05DC18E485B04DDB963426A117CA0B761E"
02034     "F485B234971B6F681CE3CC5CE58A9BDAD9267E5C075D1BCEE2C88637E2A10AD441E9B1CF4A"
02035     "324B77C6623F4DF9FC6C4AACE068F66017BC82D562722FC93B02489DFBB5560BF60F57B736"
02036     "212DD8A0533071D10ABFCB4EEAD5AD9E36E856C8ADE0B9C4F6F5116EED43A39AF8FFCFA01B"
02037     "D558D7575AF0E36F2AFF74E29435778F4BA57A5D90",  // PRN 06
02038     "214AD702A657F5A17A601D77E0F4C6A67B7B8C074514548FF89A781CF2E0B5225D95A944D1"
02039     "048338FF3AF206D331DBCB3F5E9E89C7E592A2295CB783F416C4160FCFD26983ADDDAC52BB"
02040     "FDD6A7125BA4426715F82AD9815F30B2DDA972A2E814D6BD231D453858749D65EE1B4C2025"
02041     "F61EFDF8E12E50A3719899C611682E7B2AE0CC85294542DDC04140ADDCCD5DCA315C22E5AC"
02042     "B196531A3B0071DC787C2753C3069F5D302A9FCF99B71317FF2749749C6E00B73F382983E1"
02043     "C91BC8115FEAEFF083434DEC7E8A22ECF2D1CDFC7B7ACB80FE0DD1EAF7510F694961E09882"
02044     "E50994169BF60C47B95BF37B5D6F13CFE48A9405637F87C157863C58E479488278B918DDF0"
02045     "41214BCEE2ADEEF4242F55DE971E1A7F2EC566B362F89DABF163A690C8A8B808ABC1217C7D"
02046     "49933E30272A08689F7519B35FF8C1CE0F338D0B75332D163A5DCDE714B821AD3B8393A174"
02047     "0B6BE4AAC6008EBBBA30B8E5FA2253DDCF16B3F87B4FA47C1D36AC86E2FC39DE8BAFA79E7D"
02048     "58A3A6CD83FE09448EAB6020BF121234B23A72BB7DC02A2E235BB5DE3B2AA97D3931CB8001"
02049     "65B93851EC81A1B4B7781F23E5C946F035026888C3D8C31170787F78333FFF8ECD8835AB5E"
02050     "9D165A8B1D97B19936940EC4C74D115A34120EAC800E8D680B36D9E29180BEAF5B3CFCC391"
02051     "3EE7FFDC37AC881C441F79A39F4A4161F3B403564F654B0E7D8D5415DEB9A6E11F84531442"
02052     "1CC59C6A90826974B4CC477C280ADF4FBEB368BD848AF7F91D12E9BF4AD3E9587DCD7A280D"
02053     "EE9A7F2B6F2D2EBFF7EC9454173B748D2D90A6AA650D3BB97701EAD27D6AA1D05B817603EF"
02054     "1026E643C8816F183AECE863968664E01D6AC58C2A10AFB51B1A19A5B7BDBAD77D651B78C1"
02055     "41AE70B074B9EF74BED349E58CF8A80A910F6DF8887202317BA36950F3B7EBAD45E6E94092"
02056     "1043476B0FBC136EB397DB44445706F032FC47C454BA08DDBB8E71D1E7D9045655E1039C3C"
02057     "3D199EBA21596335BB04078DB6B39A4452662D282C7AC9B76D92E112B0027B711151C56A64"
02058     "59FFC70BFFC86941736EEF705E8DF5197B4D3AADCB31E421A90FD5179D9C3EA3EBC8FD7E9F"
02059     "4C29D736076723E2D4146EC325CFD616F6B0B70AE0DE179FA8F33A2536DBCAFAA2BC0A9740"
02060     "BBF41134655EA16438BAA5F8FC96256A6F9999CB1C57B69087C0612FEF4E6CF0BE5D1FB772"
02061     "58AF418EE0B90E8AF3DC38B7A15FF6B8D2BB739FE7D0099AA4694781E581A8F7A49BFCD102"
02062     "B7880E97B89880FC41D0890C33481B01D9BA991BD41B8D6501D34E0A401474CF89D089F3F6"
02063     "A189ED80C3006515CEB7F9DEA85D0766D2989B88280FD46F7BDC7C32C2294B2A1FD435102C"
02064     "2D92E85A03775CAF11D05F2367B57F651C7586FCD32FE920B1B94EB431AD2497E75F4E3C40"
02065     "8A0FEF89907C6C830FA14402B1103DA4607F3DFAD625AEBC235D2392954B8D3BAE8A6D02CA"
02066     "E6D12EFC73EB46EC56E13D2729787CBB5C64910347AEE384972A5716B5DEDB99FCDDD5660E"
02067     "9453900DC1AF1B76D151C00E8E4BAD38E84830EDE92CC3099861C92BDBB59D8DE31AD9F4B8"
02068     "89D480FC8DC70382E8494E1A25D2140740A5F327D51D0D3CA4259F1CF5DA3474FE28270C9B"
02069     "4E7F8A714F7C69D1F65B48D7C9C69A39CC58E2C77D7A750CC0B0E5D54B2F3450B66994DB52"
02070     "4F15AF77CEBACD3CC611B2C95F6C93EDFEB652A70748DF1788E6FA2D45AF171C1AB59B0B2A"
02071     "160C075AA4A92272BC0F22631A24EC94C5D38664B08B46DC96B448C351C77F81051881055A"
02072     "CE0F364C80372EEE5BE7C660C7406B03C8F5345A04",  // PRN 07
02073     "435EA61E7DE31409D04501BDF0CFA57A84C1387E0684538E8BB651C7CF4EFD6113EAF9383B"
02074     "FEFDEE4C702FD5A070FAA3DC7D644CB3D96C8FF65F18C88C43A2E66EB78FDD1EA91E9CB519"
02075     "CDA9EE1806F0F10653F3AACE782810776AF25C9CE1C580C381D00B95CB02AF3F9892B4D918"
02076     "FEEBB01E7E730079266C4FF0AF0EC305D3A168197158B5A0D4B1C1FBE4FCFBA947A5CD2E53"
02077     "24D0C4182069C6F328465EC2900B468E145C0F41D724D65547808BED468256EEBD76431E71"
02078     "5A4ADAC80D5314ECDD03177AD9175EEA9D9149128DE8715534A648069F5EB9ADDE486029C6"
02079     "9C81B1ACECE89FC468887962AE2C063C5EC7B594D88E56F2F3FA4A20963F3BC730E74923BC"
02080     "2DCAED6EF38412E9094326930D2D2E3A0FFAB2217E9059A88882E2AFD3E62FA4DFCB5B084D"
02081     "0CE53655539BD0931BC063CBD4E660A0B1BD9C7D6CF2836C947CEF2C07B859686424B57FFD"
02082     "D0DB6C34DEF824D835918505F36755C298A09F935A2F95328248B5E923E55AA4C91BBB0733"
02083     "E4940A34AF5226C21EEE9B0E0C3D15397A4D8C069B94C2C38F61FEA2077AEBC1F7C5E0875B"
02084     "9A7F2041BAEACE95807EE05A366016B6E0FA2E7E722BA34D3654DFECEE78DCAE199BB44E16"
02085     "4E61729CC7A924CA6C7A6C061C6DA0936B4C70A02223EDA01D7B8C0D2E012F998C3B94D138"
02086     "7B6CC65DDA36360ED765F0A23EA562626E12145EB2FAC0F49BB0DD7D1ED3E9B2860DE1C54F"
02087     "EE8D60EF56D4382497C36181E864F92A95E76CC516E2E1396E2B0D880ED6CD14D2410C9BCA"
02088     "159F57DC9B763ED7CAB73F9EA5D42F4BEF91977C114900CB0F249B10DE8F4FBE220181D4FD"
02089     "B5D836AD2FB70F06869DD7F197F121FF748ADD6DF6C48ED9910F51BD7528DB60FC5A42CC0B"
02090     "72DCA6FA66B617F12FB2A9530247B1709DFA7729A6C9FF5B13D073FD7165033C45B5B7C7C6"
02091     "6045D1735979132C49CA4442986760CF8F0FDEC0C0BD5E992641075C76168B8316057A993F"
02092     "513536FC917BFB783C28B2D4878D120889945FEBAB84BFE4840C61012043E96910A374CE33"
02093     "B45F461B5492E2F96F0EF91E42D7DC755F37117852C4FA99B106F5CB5C70BE5055ECEA5DF9"
02094     "0B517B08252DAD6F20013FAD50F47A1682CEB37AA0DB64E4B699E126CEA7034969FC5468D1"
02095     "9455A1F2BA4011AE5CFFC2544329E268326345C9970756F8712E7793301E9D3C92787FD10B"
02096     "7637F5511E27A16B8DDC21C116F13B0F27FA2E5B6DE60837F011937A986C00A4D9551D37E9"
02097     "94DCA480493DCD5C4DE6F40D0E25349284131B9E0B3B6B9BEF6B9DB6DDBED89CA329D9F3B0"
02098     "32E2C05BA9999CB02D7896258E3C4504643DBF49A155F96A69148A51F243CFCC3B2BEA7E8B"
02099     "D534308E752C675CF478A231CA99C2EE402EB5DD5BE2AF5F4FAAC783141BE8D2A3BA44156A"
02100     "3D1EE98763108B9EF4670B25569DFF3568FE0E183938FB150FB79F7F28D51C8CE40089C06D"
02101     "6A09F25845B50E64AD231B367C280883CB7FFCDD5E4C044BAFE6C0915A0C52BB1487608826"
02102     "7EB84A01B9931C03DC799995B6969E65CCEC63697F1F0E146B75EB6987550C23DF2793FBA8"
02103     "F04D019C31AF4AA747F3BCF00D6FC86CAA707F1F27DCBE598B18ADD6BA048B2B174FFFCD20"
02104     "F93DD2F8E860885A15CBC2F06CB10537C205E143ADDAD33CE8CE5E6EB22C6A917172BF9620"
02105     "80290AD6FB88A91C7B63B64047BBCB60309C01F15271CAAA73AD93DCF99400BEE17F6F771A"
```

```
02106     "FD156CF28B788586A18C21739AE5BFDF8F6B8FC94A9CF1EAB0399F9FB53FE70FE5E4278AB9"
02107     "09B2A00F5A181F67CA7F7E046209A2394C5583CEF4",  // PRN 08
02108     "C1A7D5578EE6929147D34AB5B29B97864F46686736A0966EE7298963AE27040F4598E0AEA2"
02109     "35B562DE84BB1943891101155AFF6D185D9F66E5F9DD037B048914D7C1B55505455D08DB2E"
02110     "0C58716AA4C4E779EB7348D486F445CB8BF499DA3054F087ED95C2561770335DBA67F1EDEA"
02111     "C600071AB5918ABEAD7E55C5C6E609896ACE2426C47941C1FBDBC4804D8FF4C9290E8ED35C"
02112     "967D65837D6E2B153A3DD843712AA34BE1FD7CE0D829B04DDAF036D9D16063E28F627491C1"
02113     "C37018FBD1EA3D33899BEDE421C1910C2957C13B8CC4B8A06C8D3B2264E8E1B8FB5811622A"
02114     "9986B4B6F673994F6F5FA072ED8A8564C2AD450CF801C9992779F292994A98800AB12A9EF0"
02115     "7F777F441F19FED6314404A9949218B2ED4E239EE677D8D1425D0547CBA99C1333FCB1620A"
02116     "8D11DA5CEFC51DEF5B3C4862E17A6BE3C194F88B7BBCED98325D638877757ACD448C0A970D"
02117     "63603B58E3F4B8A361D36EFCB6AAFF2858E14B1830F0FFFC4C5CB08BFC21FB091045B9A7F8"
02118     "DC0FAA02A62D1FDF7178B5DC90D2111A64CFAB0D83680196B21D75EB91A5B3E2568C7E8A6E"
02119     "DC65048C88EB34F6494CB697E44F895012AFCD7AA86DF341CB9C63A7EEE8E5F95E3D60213B"
02120     "C23C5A83CFFAD4225BC731F07B34FCA550DE26E0D6DA29F03A1D9C4233D9A0633D31AC5F55"
02121     "A11C1208C8B85306CAD41EDDE3FF5EC3A61194E724EE2D6A4C94FCDA217BD7C7FB328A9CEA"
02122     "8064D94FC66049B56AA1DB9F977216AB4242D0EFA59AD18540F1430A6F4EBEC5CA7EA81EE4"
02123     "BFC6E77F9EB6352B71275D450843304F073AA82D77EE5602B43DB62B9A733320A67B2727B2"
02124     "1B705B86D5DC1459B5EB688D37236FD37A7163E78E8425C589DC600609DF3E9F36F9094D06"
02125     "C60731527B8ADF58A95E40C1390FF657D418D675F528B3B665FB5F3C56504DF40C1D3EAB6F"
02126     "3CB85F5C7954587F87744775382517B4C652B4259571A8AD219A6517E2D6D3212AC1684404"
02127     "DE7CF71478BEF7819DA24B09A11DDAD5717CC63995BF83DDA735C3A50F6B164F26D6A79E3B"
02128     "20A1488DE2AB44347694DB8EDD3D957B6B5510C4A266E514164D23F1594D84487757EE6F43"
02129     "53BD06CA7F90912AAF5188F217FCC11A62198E2DB04A016D36A583891D346C0D84241027DD"
02130     "E80090356ADEE1AAFE7CCDFF983FD7F8827BE286280BE118FD9FCAB26A10D19BCC63A9AD9D"
02131     "202036193CDA36321A5FA7252D0E65FC0D8E588D70C3B629D830CD1DE2798309A6B6113FC2"
02132     "A16B341BCA4271B5994ED767F2064FCA7DA335BD1D6B21F54CFA0DDCD47E2A38CA924DF85D"
02133     "2DA869F6DEA8FDDDBD20728A21034355728DDB4287FF0EB5A9DC113A71F7C69899EE76BE7D"
02134     "186B39C7C868E78C03E923FF199246908FEB48AE45261B3F7A8D6E4FD0C280545A9E3A5789"
02135     "16364196E483F57EC37439B0FC9FC37C5CF56E21A5CAE0B70D25B928EEEE456F19821DBB97"
02136     "4781C646E860EB3DDA8EF8577CB0E3B9E96245E14CF4E51A5D11F7AD9A77B932467808FF55"
02137     "E588455F810CD50367DB1621469AEE8BA6FCD52C75BB027521C530A918F9D646C69EF8DEA1"
02138     "F7B14354815F11E0187F01F677E2A0FBF78ECA2B1E51E898E26CE55693ABC8812F4CB4C10B"
02139     "9BBDF385C6AF0B340DF8DD2E957D877DC1AEC30D4A6195F40F09B4CF4010E810C764EBB578"
02140     "91652C156DD20584E427902DBE82AAA3CFA728059D31C36253649FDC00CCD29517AF2FFF4F"
02141     "5050B0FCBB73B2888C989761A7F0551375E4D3C54E0C9F8A4679E640A73476BCC6587195A8"
02142     "2CF5E6F550B0A85D57A97E3E5D0B9A194EEC3E41C0",  // PRN 09
02143     "F0E94A299789538824A90801AE02D93A78EA914B9A8F10FEA2D2A7D0812009C732665D48F6"
02144     "DD58CEE4CFC72513CBADCFE171F837E79B8776B392B2224C02766D0468E513D0BD1BFE6ADD"
02145     "6C04374604503A3EC8539CC707110D18FCDC8E000EC1A3DD03F7008D37C1FA02E419FD4FC7"
02146     "A6C81157E07E353F3C4882C2AEF4A0CCE1F7A5CB5EC753528152C521A2ACF7548F19862EC3"
02147     "1D8783BEF406EB7F756E41CAC3FBAEB9A901F9268E78E06ED5D6CB19BF4606C9DB660C03A9"
02148     "D5B06BC235DD7936ADD742685772567F72C7F36CFD7D4D09D3726F1908BB3277AFA69C7166E"
02149     "A2F4CC9387B13AAA8131D9F21A5D97C13C7DABEBCE61D72556466093EE0F87E9EAA9CB4224"
02150     "5E1BF55157013C4350F38596CD26D2819F98698CE98F5D4023BDB3C974B0E61FC6E6604CB1"
02151     "707BB22FF5689F3DB3AD4831BFFD3F93C702E0DDBEBCAF626FC2655447134570B7150E030A"
02152     "3524A69795792960CEE062D18CFAFFC4474490DA81268B27B60E9E9BAC2A9B7D2E2377A2006"
02153     "AA5C439C263E8F9EE360B42E217646F7C44EBBA6D9E3B98AD20B1BF7CB8CDFC62CA7AB139B"
02154     "C2DD1C85A837F51AC9BF08B3F36CEF176A03F358E2E37EB5DECD0DD2D544E58D203910613E"
02155     "B14744E914F92C5E9274FAB025CB6D8AF16EE03FC86AE3947F7A30FD8C1EA23EC7EDAED405"
02156     "4174E4E6DE048C563B1832F8839927ED3EA54B19BA41CC144C8D8A6D8FB66032BA6F4E16DC"
02157     "81C98D37129B3458C2F2A6517694FD8816E8C71E00D77189768C423702612D9863FC2DC1C4"
02158     "593809C74D0AA881DD7477BC584376DC6C177347EE17F3E17B524D0DFB315196035FE6AEE8"
02159     "3675494E23BF2433310D4C72E84DC82BDCA9B791D5BFD620212172AB2562B5C36DCA28C5CB"
02160     "8078CF41F2BDFAEE651B04243326DCE0C51ED88699CFAE4DA3B478DBDB92DCFC8D2EDC4472"
02161     "691173B6C5B9E5E116C30DFFDCE7B0381E8E1013CB392056E3BC7442B2A2EBB631ACB65DE6"
02162     "39B3CFD8EF3918531480A35314F3F23F9AA6CA41A9B8F6D08F6C589DD2584B6B8285230808"
02163     "EDBCBB99C0E918D4E73AE7B397328502F2D47078CA2FC0BBACBFC9D9E491EFA2A2DBD69D82"
02164     "E0D829F74D5ADEF4100E6195CF3D7AAD2041078AACAB8A8EA62FA42C6AAB6203D92794F77D"
02165     "4E30B84BD16518A9A0BD0AE4B372A537BEDE43B31A6FC1A58F9B43AABD485C55E630FFF8A4"
02166     "359DCD52552B8A090DFD5CEFD972551714E5F0A2A16442345B33DD9D232AC8431EBFBCC5D5"
02167     "165D12961D1E954D01C0379B5489AC0DC0357667325AA864D44269799DB710BC6A25D90AE8"
02168     "E0C9368BE911C91D68A49815850B4B29708BAD74B4BF3590E6488655622DAF3A7B52356C80"
02169     "44871332233376793B33E583B622EF873962E8FE259DA8F5DF61748B6799A14ECC763A1B545"
02170     "42D67E57BB73E411AC023907E54BA0EA8B7F20E55739BEBD7F83E408238B73249CE69FD2DD"
02171     "593565141FDAF12BFAD3A43670935354ADA4067DF0AE4150AF306AAD3071EC4DC0AEDE6F56"
02172     "8DCDDBABEC98E4D17AA99F8E881600D35B25CDBDBE5DE184180A9C35EE8B42F6890E068700"
02173     "F06E4829A742FFEBBB214D17BA925B5C0B41244DB5BC7AF49D9A24B2C135131E553CEDB2CF"
02174     "4E6E2EAB840AB377883230241238 53A19CC04F6E344A175A4BF1A16776B7F84A17501DE4A5"
02175     "94BC515E788FD0AA08ED4BA3602838A6D1D485B1CB80BD24D97905A9A7E2DE71A9A4CA3268"
02176     "9C46AA9EE52A9EA985C85C9289ED1577B577C444FC73FD1BC3DE03F9FEFC10E6A17553A7E3"
02177     "FEA6D57AB71E2379C9C5F8904EC43E57D5FB119CDC",  // PRN 10
02178     "A8C2390D6C3F0B9D01B9794A4207DE3F62F00A4D1188C727B2910153D9BC6337AA46D2D79D"
02179     "09DB7656C0B28654F1FBAC5C3DAA8A75B16BEB5FAD8A256E38EFD232E007A741028319EBA4"
02180     "A6DBD0AE1C1F8989246476AD4090F9F4F560D0CD72815FF6AAC29C8674B1D366120EB32133"
02181     "00E99DF390B3169A1CE03D2A5E471B9B8DCAA957259311B97817088B486CB680047359440D"
02182     "D39386DC2BC24AD2AC30436A08C335199F08861F93177640074A241D9C5884EED07EF0C0FC"
02183     "E8CCE08BA30221150C9EDC6562344CD3E14D3735B41DE2CBFAF4DE0F6D417299FEB3814662"
02184     "5F3C70D8872103FDC6970D9D37D1CAE92375F81B26566A78A02139CFD83415BFAA5D328674"
02185     "797DE51E4AD9B3B9E1F9BD18A0FCA255CA8AE7E9E192ECAE5C60E9F5D578E59F7F7074A12F"
02186     "A75F9388CAC0A148CE9223681A93D62A679254AC2F41CDD871BD73BBC6C4E0CC9DA67633B6"
02187     "1BFEE815F4E15ABDF639899DE105A5F4D74CCDA3EED32219BB447EA5C6411CEDB351DD76CF"
02188     "11456EA762C2F300671A12DD6ED8EE9AEFB4D0BD86D0CF35AEABC9B9E7822478A6DFEBB887"
02189     "DA0E2D3ED4C010E5031C4332FED83480C21ECF904B65925C4CF64595F20D072C7C6AF12DD3"
02190     "C76C682833CC6296DBDC73DC8C979004DE4ECD749BA45D9748DA728AF757FF5422B8622231"
02191     "46AF86AD4AC4D4F407DC2025B8166E2CD64C8E1543E046ED91AB1AA5524E88AB6851752AE5"
02192     "A2F02F933DB3CB7EF48775017E000386805EFEB8CDCEEBABE613A87E0C554F99DF0C8FDE49"
```

```
02193     "FE73DFFD48379299FFCF738EE21B9AA1AE4DCE7BAB98A356241545B6F7D186AB544F77CB61"
02194     "B6492BE1F02C4B77DCD5083DE4319D1A1B792961D11D552F8AFA649EB5D2AD64227524722B"
02195     "0CC97260A8B3B6493B00D258A570CB238D3C2511C0582DCB478F878528B8A88A5B33F9FA6C"
02196     "41302DD1BFAC664C692001F7F7D207A8850D6FAF22ED887589405739EA5B5B9812BAE61788"
02197     "9A225FB2E6F9EAF77D3B3758F9ECB89637D1FB20C428AE3420C3679B834EFB0F89A389FFD7"
02198     "2D7420F2435D118045FF7B7361FFDA0BBB6934F81AD8513A626CC25D9A5E2A5FD1C51229E2"
02199     "D5FF803B8983B672E92A1FD72C16520531C694F44C9C99BC3385A610239F6963E0BB66ECDF"
02200     "4CA29E1C175E414A5D3F9E9F495F2D03A073965643186C46EE9BDB41876A41ECDA224719F5"
02201     "C76AFA6E955061805F023498AAE965184ADD6F6BA1DFE0BBEFC3D1EDCD40D8F55BB518C9B4"
02202     "5CEDC95682B24F2818795CBE5D8E504F0885863C750524D9D238E4110AEAEE095E1C545C29"
02203     "89BD7E0BC0BE8AB750A7D319AE69CB080DA562F7DAA7068F38B14AD7282E041835B58E2A48"
02204     "A340F993D10FF4F805280C183A3FB45555B8D54EC145CB7B9BEE71D7BA652FC533ACE1BC8D"
02205     "621F7257422B363A5AD2798FB0E6E536854F2BD25914AA0B6DBEF1E2EF71FC7890E20FBCBD"
02206     "AF751CCB0DD16098D96FF22961B6141257AEAECACA339750950F6895107355F04F9BB3EEAA"
02207     "B3A9024E4CB79B19ED7947EDE79958B669077E6BE182020536F7F234AB2332C2F2790DD96F"
02208     "5816AE3FB5F2CCD72C6D177845DD0CACB44128F6EE6B1C2F673A8A50842E13C8FF0087CF71"
02209     "B8BA0715637A40021998652FEAEAF5719ADB71EEAAFC8093E525602599278B35600C743AEB"
02210     "41EA78CB0427D0894158DD4ED564C4742033324C58B01C15822B97C9634FC8F320C0B103F0"
02211     "A904D98C9AC529FA799B30F8B2C0888E47A3BACDAAE5A3DE6DF39A693A9AB3D95E7C973C71"
02212     "25C0F7D8C59B37EC924D7B9737724DD92818360F2C",  // PRN 11
02213     "2EB63BA5756AF5865AE33947E311A5EED4A0CD056442E19888A2B8302ED8017085DD5D137A"
02214     "3769058F0AE0476AB202CAD822D6A781291C537120F316E90D2C3CAD13381EDBAD35505A5B"
02215     "7DDC41E46771F460E501B9BE17DF8D4DFFF631C365AAEBDCB7D5CFF7010BE2EBEAA0B99820"
02216     "E0BF3D290CF27DDACF397A5E4922EEB7D8BDFDEC96B87F8202BD409F31A200515EB6A05E25"
02217     "A5FA504093F1D82A83A777F5389B5C90E66C369BB654598CBF3F258E9E14DAE819837A86EA"
02218     "775A05EC829B4889B816154A2B7882EC7FF71BA7CDAFF873B5CA4F35011DCDED8C0CAAD101"
02219     "F7BDCE1CBE67E081D98EFAB71490AD12AB7BD9B4CB9331B6B04D6DA2C12311F96E01AEBBB3"
02220     "50BBC7311E8AFC9A6F42CD541ED40E2D7BAB44BF7F1E4465CA942A0E8767A38D89D0231BB8"
02221     "97A995627EFF71B3E4483ADDCAED084B402A3CB23D5F91872BAA49AB361281454AADA57AC8"
02222     "38AE2B34B799A1B46595AAA02A9738019EC3644C63BFDC72F72CFC35D48D83B831E6EF3974"
02223     "74C43CF665D64DC0376A437CC768862A92C941B9BCB8FABC6804C03BEF7BD2C5EEE4609870"
02224     "803A5E9BB2FBC42028C5619C3706F773426CEC39D096FAD267BF5E6C7C7510B0D1948FAC57"
02225     "3C5716EB4D6AC6EEC492B347497B47C0D18C77E7C264E9A04A4C3DEB83B1101E40A0A8653D"
02226     "9495D4A7C13A5BFF3C6956E668DFA999926231811F3AE84195B65213FFB824478911230719"
02227     "E8EAF206DD99C7CD7D3D7661E9F39250793C1391033EA99A0E31D8CF83997ACA08C43B4724"
02228     "0BF1E5E9D1D741CD9E6D3F5DC2043C9D3D68A2A689615D39014B75FB53B4866488C4B37B68"
02229     "B5DC5F9B9D6AB9AF190789CD610E939960ACF607A488964C33DF0091FA4EE86654142083F6"
02230     "8436D9700B3B50949C0D97098D969BA25622C2EC15FBE7D7C7B78C1C49F671672971E38996"
02231     "C2D77A024919FC1A87BB9BFEFC951EB75326CE48D3837068F31B9592DDD8505AFF1E0E3191"
02232     "1D1F4E311CB1B81DD8A03D68A5EC075D8ED7D69A1D595E7AAA70928DE93B979438315A4B98"
02233     "EABDC8B5E932257AAF79A9CCC74E6FF88C02C33217FCB3FA8DD72AFA1E549B9E5CF6099E2D"
02234     "ABA07C91BD3BA6A9A4936556FFDAD89D5F080DD7F818EBC8ADAFB3D51D8BA7B48209AAFEAC"
02235     "184E1A869FC88BDF92E77C6D4B8B983F4B93A52C473F1BF27B4D7514DBA522ACA79463A0B9"
02236     "C2E0D9A2D47DB238AB9E2A1636B9AA825272F2B9C8D36D64614A4CCE2EF31B67682F328B46"
02237     "D31FAA31DF874FA15799B5D53E143C9E9127AF358BA1B43A9418FFDE00CB16D32AA137FC7A"
02238     "5A1CFC2070FDA112DE600F3C05BA0F50F8AE1703AA2CB64135C29FA621385C803673A4D771"
02239     "95B4B74E554B043FC07B106533D0B41F7D882FF3702692709A64052C4AAE51CDF8B733A92D"
02240     "86D507B6FF4CFF5F3BAE4D0D9F3F90AFFD774591E6E14539CAF99C833D2A1D9A12F7E9CB1C"
02241     "F74FCB353D11E22B8216FB91564DDA8F1D2C9865080422DF0E0D8AA47577F31A7682B54E36"
02242     "BC8344D5F5F725C819D3DD3A948F4E6F6C8934911272671145DF1B286C18C4F3EFEF0BB6154"
02243     "1883E413EC3EC3BBF99D83E3919501A3CD59341BDED35BC39B655D61686555CBAD7A1B033C"
02244     "2E326887249E6E30AB12E43793DD1B72EBE858B40ADF4400A59363BC56063115492E114688"
02245     "F47447132D606F2B99DB9CA4083A91E35AED045C513EAFB7A3F0040848C1DDEC0FB3C2CFF7"
02246     "17BC4CA36C425EC3CA9BC51B3B3C1885FD8B4182AA22AD38CCF8FE3990DA75CDF5712B358B"
02247     "E866A6435329677CED89571C3877F9ACD0D3E66884",  // PRN 12
02248     "8F0A4621BCBCE418BF2A77BD47D671191216B230C276A0D081C4F79AEF839857586F9DBA52"
02249     "B4E35AEF22D86FBA66DAFE801C686F03342849B5C3A492280950C7F3B725DB4BD6984A6389"
02250     "CEC1AB86F29D653781EED847D7E69C99320F8584700E662070E9C43957F4DDAB28DD242741"
02251     "5FEC6F1503551D972DCF11BC4EC3B3FE745DA7F717F35832B8CE373A00AE6572D8DFBD7720"
02252     "E00517C50E2B634B6BC3942F9A8C035AC80161CDB45AFDD7E9C473F8EEB4F1FEE7A677E820"
02253     "874F03B546464939F3EFDDA157C9F478988F3ABF39BEC02B722C559CEAF997A13AF66BCC0B"
02254     "BB4DA7B68EB6C6E8498129A95D021B7C3A072683593C2990627FEDC52F66F3A16A8064018E"
02255     "6618BE5D7C63BC622DDD4A15A33EE19E4161A50EA38CF6626D5A088FBC5721F567145CCE30"
02256     "C66DE60CAD79D31B703B129D27627BB5AAFBD9BC7D43944B9B0D255D8F338C7CB7EC56D121"
02257     "C3F4E61FC0D46CB0E036262FC182E8F266741842DE923D276CED79D4A25FF7AB3D2571B1B5"
02258     "89E01583371BC8D8196A1C991BEB90F7FE71B99EF9E100375D3E767637B2BB1FC907846C53"
02259     "550DC4E6C7B8AD7FFB9DBB65ACB30C9644FE9D59A74B0E81A5BA068CBAEC9D6B362BA05202"
02260     "260B1E86EBB67BAE2D74F60B7E1C0B510BCE4EDD8D71E342D5465A8032E87617790ADE2770"
02261     "D0DEB1ED26D62A863D030C781D2FB02490718A5B55D1BF6B3BD1E3B3C163361B2F13E616FF"
02262     "C87C1BDAA7D7C88A331511ACE5A45CB94DBA557EC47A670BFFE839069A546FEB7183C27697"
02263     "A56B65AD762735782FA009350ED52819C23AEAD72A7150DE845B899669EA37B4A12D63BA7D"
02264     "6DFDCA495A0957BDCCEE6BE11B8B296358E92E60A2317CF4B7542CDDFBADFD18C88F5AA498"
02265     "4614A113E992DEAA18FA172C998EE6FC8E936E69953EDDAA3FD233182C7B986DF90386B366"
02266     "1C649EB0F9A7E65B90572132408929968CDC39D18020C6D81A3665639CD2A3612FB2135A6E"
02267     "057F0F68B10D3CC1C35DDA2600AD1C0A6FCC0C5DD0D82F6218F2C1F8F5B714C47636CF68D5"
02268     "361D97DD1367681116F2A2A045894A85F5FAF1BAA27F93A3B880940418104OC24156433B3B"
02269     "87467B27D7C3ECE44EB763D14356D6EBC7542E21863700505B8A10D4B4B6E9D2A4C2EEC965"
02270     "3BBC0D9FC68DCB53AE7C8B6CF3120C921B824765E7641A3E9CDDA804CF341012964EACBF55"
02271     "D6430673B83C12CDC53100DD8BD6B18A56B5AB68D83EA484F48D4641B94AA2816BF06EC1D1"
02272     "285C11AD060735509367FDD0E3004139114428460B240612C1389CD4554DC8D7D7C7E4D4B7"
02273     "9145D4684F1FB75759E2D506F9DFE7632CC553B945D91FE0F9A829AD6A9A18817B09B986B1"
02274     "8FECD615414A0FD6B6070B15F2484AE8B3EC3DA36D7F8007A7BE336B749104C2695327A49D"
02275     "34741D76752DC943897B29C9651EA3C034BC4B8E1F380A9FB0F9D2C23B627D9DD8317894AA"
02276     "754B8998573F025A024F43E03EE86A1EF583CE02DC7EC2BF2A8EF9A3C7743DE2E6DE38D72B"
02277     "5E33BC5CDFCD12184C9664EE7A3F4FBE9498F59107BF8B6DF69DBD6732AA8013FF8B4525EA"
02278     "9E40FF6F96FDEA2AEC03311D6E8908D426293E369D52F818233FAE1C2BA2657B38FB82E786"
02279     "4CE1177D5E21EBC8AB773EE928EBC4A7EE9F4A9CD2D9F15146C08E4BE0A77CE272A98AEA05"
```

```
02280     "A6DC18D5B856EF1B402F9851BC84DC95FCE71320D2A5540D40BEF67EE2709147EAF3C3D5BA"
02281     "BB0B21090BA4C94CF7242982BE35925F691847A7C296B973EF2172F83AC9ED5E0A0C91AA69"
02282     "7305F79A3FC6A8EA4E7D1DA88A81D4DDBC1F604BB0",  // PRN 13
02283     "896DD4C837F11BF155D4605D46411A63FECDCD64D7AAE873E1DE22DD62CC2F8982EAAA66BC"
02284     "AA8DE8DA687910A03200FE437BB87CD5E6D17C6F5B8CCFDA073F29183A1E12917C5D78CC1C"
02285     "CEC5C2B04040D39E1E74867B21192EFF2629AEEE22427EB5D2399515D892EEAD80934668BC"
02286     "A906109F6ED85AF29E702FE6C112992CBF851232A4C140BE6CA9C8DB383D1729B41BE93970"
02287     "4297BCB41E4B8C7DA5B24B93FE7E7D2214287145A764D3263C9EE136457A2A270EBD03408E"
02288     "CB657B78CB8C0A5C1767521D75A8E12D9FCCB3CE8F6321B55C50A6102381E95C12644549A0"
02289     "E5AF29183316016B89F6797BD89BA8F1E932F4C91C0460F35EAE32F46D42D66F639A87C25A"
02290     "C6E9B9C4E475B7A4A94F10EE0C1043159CA71BA3FB4F794E766C5A7C6F7B85088CE1A2584D"
02291     "4A765D7888F1496ABE23ACA42FC001E2D30B3959AD7BD941916E27DA648833EC82004AA093"
02292     "7723CAEC0A8E7FE3C6742DF8E1CA797F13FCC4A07BA75874FDB6A66014C0EC688C5F5E0D1A"
02293     "CE231B32A1A14A2503155D412BDA98FBEB450D35797BDF96264D141EEAC77550408F6A6399"
02294     "3DF3F89D1EDC90DD0885227C273546C2D86D3513AC3241AAF96C137B69970E273B73EEFBF8"
02295     "668B9946CBD118CD1977445F87557DA03668D4F3A6B5DE6D0088982F6C5DCF082207A99B37"
02296     "6B155FDCBA8A5E47F2CA4A50D48DDFBE05F8C1D28A080147FE9FE04FD49C370DD9E4861D8E"
02297     "553A22B7DF7324C51C20F10BCFC18EB59BE931C69232419F5E037333B52FCE4A138617B087"
02298     "67506B793A430CB3EA0A4D9A2CCCF813625FBD2F471C39DEB2DFD0792F3374A980FA4B2081"
02299     "85332FC68798DEA1728054F28771E0AC3FC065645B3E996EA729DC6A29CC23F872BABE6A84"
02300     "911B1B6C256C38C0A62331EB66E2F8B9A77ED79E323B4A96728F0ECAC851C3E9C15F2863CE"
02301     "6839D3752F4F206C72EF0BD60F087315E6AD3C7C836FA3C5CA6BF2B72D877E33C17EBEE283"
02302     "EFD62F395637E784A84A3E4EA174BB501D2CC216306673B06433A05ED052860666A3C3D7E7"
02303     "D7C991911A56100A31048BB81FF71110697C941CD0ADB08705A5A73CE66AA87008B08B51DA"
02304     "BBBC3FCE66B11135844BCD8CCDF4D010D7B135AC764B678CF307154B5A96D1541B75FBA65D"
02305     "7A0D5C2BA2D1DA48BF0EFB9BEFB5141AA6D30A59695EF7FFAE250AB52DF0969FCC349E1C7B"
02306     "8D90099EF0410A861FB05E37D7156991FC65542B3495CCE223CCBD383DD4432113B817B0AC"
02307     "4000F9EEC55EB377CF48B0F51B8D02359449CBAAFD6AB69E3616D6A3806BF67069F2572D1A"
02308     "A0E7DE28042DCA01533E1C381449E22A68D901424498DEA67F0EE80FAB6FEEC56BE0195096"
02309     "9550F01D62ADD52396E927962F209B262674D2F079D0E48D1C994864BA90CC4CCF3CEA06A5"
02310     "995072904CCD97313E82D3CF76F24E3B3946D11FFF45319DDFEE619D2B5D777795567A4476"
02311     "7F12B41D84A4BF738A2C6AFC0DA08356DFAC343EF090176425CDDEBD1264871CB7AD526D9B"
02312     "4E469E866817A1ABC9973C598B1724C8C3185DA3C999636074C27320694F0BA758B0842D2D"
02313     "3EC93DA9A9CD3F65B321E378D95F003C744EBD7BCCD4712B6FBFD084D9EEF0C84C70651E3A"
02314     "D999D64BCFB5E8F7D9582C290CE59176072A5157B1CEBB5684D47271ED482B9FEB7FDDE78B"
02315     "424988E88627FB1299AB8B96CAA72239BE6930BBA4396118041D6F3CE51CEAD81DA54CFDC8"
02316     "BBEB0F6C8E9FAEF93E2C404D543A0F0026C291CE992CD71BCB0C6E9C459143D995C92A4A1C"
02317     "9A6AFD36E1CA87E1F8EEBD30EC9D209BB3F6410018",  // PRN 14
02318     "0245F1FF6639D3C45134190F729CD06608426830011170C357160EAF351BB00AAA61F254B9"
02319     "39D89335881D5D15F3DAFAE92ADAB2B59DB7D3EBFF526D9A6F44E1D8DD779753433A9A32A5"
02320     "41A798805AD307787C1832385D8B11DE57BC3ABA2A81B70AA2221C8C9247EFB3A40943D8A0"
02321     "677F1D4E6F9149D97D6D83173FD3D3114293D628AB97AA4CC0121E2A88BD5B5C290AE7E5EE"
02322     "4BCBCC1CE73017E3C58FBE575701C919ADE5E94208789CC2BCC330719F022C1F79880B1BAC"
02323     "4E377D486ED467BC58B13F2FF97454BF1B4F4DEFEB3B5998FFFF604EA0BCAFE859CABA133A"
02324     "F969827887AC252B9D21D8AEC49A0D77FC83B4D9A2930732E9A22B7B823E302E469B52537D"
02325     "B5341AA0979642F4505151AED71B690430AFBDB09A8454183A23F86F598719B86D1E3FCCB7"
02326     "2DD6599C9895CF6AC28FC5B1D4A821A6F9DD9BB7AA63DCDA1678AD816D96B919E47D0D0E95"
02327     "756BFDDC71AE430C4F71E521701452709B6E670306A20C2F60F84D6009DF6D749C5E02D3DD"
02328     "5495DBCCBE9326D7D302E2A56032F4CA5C82A107E06A867D09A2E5DE7DB9C250E25B5279A4"
02329     "867657AACAE0E3D601DD3BE26C936BF527CA9EEBF3B247F75487A01694C7595619FE6E90AD"
02330     "78BEA799B0BB0C126ACF3DE54F805725CA01879BA5B8412D4FD3D3CBBB37AFA955CF90248C"
02331     "C521B0FBD0214A8D4C9E4A41705714C8DF5EBFBBF2FBDAB93707B0D779357E3DBBB20D5CCA"
02332     "E707BA7BC5D21D4C23D72AA231E83880867FFF3A6B587EDBF5485C516239028271B5462F78"
02333     "E3B226B057F01DE2D348C7D08108B990DE8484E839EDAA174B3CA0DD6D1338513B3D1E4ACA"
02334     "BF9C14A9E03CB373D039645B6447D7853257F74665CE10EAF27DC1640C0AF88FF4C207994A"
02335     "A23EFA97D1CE707AB08E4A615C580E3AB971D64371715CF0BCB8421D8178D708E4D3DB24B2"
02336     "AD3143E2176B4BE37DBD8812116E134DC9B72E35E8811F7F404485ADD7321D3DFDCED23BC8"
02337     "209A9DAAA198307EF1BC0E204CA42D7274EBFA74C88251398D6E479D53C5C6E7528BEA5BED"
02338     "C8677F6F5A81A07BF7CE58E4AC38387824300D14FFF131A54390711F95E03CA59DFECE3EA4"
02339     "696643A16874F98B76F6D280F7328AFADC455C51712BA748D33741998838D33F65F0166BD4"
02340     "D9788B129CF7F9F0802758631E5A718A5058C63B0D38D9E6877778FB8CD8BB7AF099ED442C"
02341     "52DDE7D3C0746BBCB8595C35C815E80C71E98425FDC629C7F8084BA2556CEC31E707408BA6"
02342     "FD7A2F0AB83286A93E565FDAA34B7344474EF48FE5C99017887C8086AE4C89169141421515"
02343     "50B0222C5FF617817EF86920F1FC00B9D9E5BD95C60477313373DB142398EB7E5F1B29EFDBE"
02344     "41ED1E9BE725C2748EAA2F7A1471AAB6C15BC4596B6C6CEA5CB3C9F1E57179F96B3C3017C2"
02345     "244B179C384ACB93557170F4E0CD387D1D3428A7030B4276F8D3FD38074E9348033E3BB3B4"
02346     "A05B3366064C68E0B25F36C2E45FB0AD4606CF5A6B8FEAD45CD9790BF5BA1B0AEB5C67196C"
02347     "05D637E878F6D3108B36ED2CBC007F6BDBC7D74362085D651ECFDAE47FFB069ACCC7C55B72"
02348     "6C0C6ADCE2760DD3509649DC58189CD7195B11245D76F64DF2D61B07E0C172313F39961BBC"
02349     "410E4000C7F7BCC52593AB7915D5B5B3E746CE8F15AFD2AFEA8200687BD1E0D767ED297347"
02350     "A1B0634A4F006033322F787FD901E7F2492E1B491D4F5B901FF12CA88B2B2B2982D7EEC9DE"
02351     "C4538F8E946172AC82CD468812EB9E7062AED4075AF0A587A92E9BD971637B3222CDE1A896"
02352     "5DE1EF2CD4A36F2B7457D27CD328016FD2D92B0560",  // PRN 15
02353     "EB01608B4B6324D02441F4094E82D3DC8ED644FD9D7FE96DC544276D7E02568453EBE37597"
02354     "4990442B96B85A9DCF163A5974EE96274A6AAA9EEB0C83AF9A2E27C4A4174EE1D4A4035782"
02355     "AEF83774E93F67A5A7B91E7C8E05611BA6747FA9F83A788DA50C9D4EFA17BCABB0361E055C"
02356     "E6D07DADE204EF601AF3D1C7A54AE1011C2AD46C27FC060A27073781A221D182D4AF531D30"
02357     "56401D61024A29FBB0030B8340E91AE3A32468CA48183567199F34A76209F7846ACEE0B5DE"
02358     "327B3A381087204DBB6FF40E1463CC7BD38311DDB99FBD05BDAF3B84D9E64958F67421CFA0"
02359     "9903400DC87E212FA320DB95DF83AD6A6E247E92151AEA0638D3755A6B409E5853E1A7FEB5"
02360     "4FEC711D75AB9CAB21DDB34D7115174E01E9F0A3E650F852666D06A2A8E133C872FE9034BF"
02361     "9A61E29CC319255B09E40096B6D15809FCE4C8993A056E4133BA1FC05933B643B5D476D398"
02362     "7E7BD372412852D439D38600BE4A7F22BE1777CD86F70F05DF3B260B41DED95653243419C5"
02363     "BD9B6288697460A0FF7C0E2928D707BBA41E46D1AE11BF3751852DA1516941BF67E5C17753"
02364     "83A7F88CBA065BABD2235C155FC69D3C11F5658B019B089A9B3A983A2D2533BFF7694FC271"
02365     "597C42EF35F8AEF804E18AA37E3F42E49D23FB9171A312A562BBC9B093D92D45EB6A1A8A66"
02366     "5257D596174E1FC44E117397AA7C4C77FC1BB176C8284DBF4558B5259FA69836B68322705C"
```

```
02367     "F783121E3C4513EC11B76329F1310731CEBD2CC8B7F49AE8D87BCAC07B1CEBABE872B5F83D"
02368     "71FD10F300F9FA30F4054C476F86EE75AC68F1F80E352D3F9753233E53F67C3917C26F4B0E"
02369     "8A61AC0299FEBEF95D9A46BBEC9F5C16B5EF127EA18BFA609B99B0B362036F4AE14D11BD7E"
02370     "C8ED2B02C8BCAEE2CFE409899E06DA304E23305DB1AC0A7746A72746C24E6D073F1FFFD3BF"
02371     "AEF300299C8009043FA392DE2054F237A098EA5FF0E537105F25B7D85355A44FAE707B0A3D"
02372     "1D6B30356CF1BD6EA3BE57B3A6147B943D1C96C4E2C871DB1AB64D9359F1F3D15F9DE5F45F"
02373     "E8E03269EBB9759AC78F47DA7B913F09402EE65271859D623908D464B1B5E59E4580BE30E0"
02374     "1C553DCC79B190A809C71D8204A735350C75C3361EF3C89C07B84270502BAB82FF08979383"
02375     "8B13B3041A17AB396C09C4BEF716329318B95C7C4779889956961F8D2200C163C86F45C1FB"
02376     "A0567C8615F4E815AE8CE2E05589C006B4560D66A67495C2B99E3A951500FECF953D9FA7FB"
02377     "5843F102FE0F82E72867A65C09318A52D4634A3FA28B032B92160CB2E7D6ED156586D071B8"
02378     "CB147D139212BC47774CE580E80791C14FE49DE7688B3A99E966A9F869B0A7421062F18705"
02379     "AD23AC238B52E6D80F43DCF2B43F66965F6075469E0E789B40687D104DDB5CE5A12C0B741E"
02380     "DBF1A3E57BDACC92265D592FDD63703E534B78DBE9F13CA04B9920EB740FB220E01DC176C0"
02381     "9CA51EB905E32398E739C6A39C02043042C5FFBDC9BF959D56842DB894FC108AFCA51039A9"
02382     "B28D19D67127856E29FBA3812F764D31AD4E2FBA5F00A828F85087B4A5ACF24CFA79A3B60C"
02383     "954DCE40BA6BFAAE4466C88A60293AB46FA2FEEFA2E3B0DAF0CD72B0CF914848AE6A9BC26F"
02384     "6578D08E896479879D3469E0CF39E3EA6508A412853C83BD33D0B993678FA08DF14AB40F41"
02385     "CEF0E7A4B992DAE6F884E193B10E5136DE079BF85936D3B5A586323B08FAF8A39B24ADDC2D"
02386     "7E1EAE1F2F90A39F26B130F060ADED352AF085F9169C4E21093FB2256C4E17416F57D29CD3"
02387     "7CDE3BBEC8CBAB280DB69643C86696C3A6D3517150",  // PRN 16
02388     "ED28B364884EDCF3082EBC5FCD11FA0B0823DBD808FA19BFCDFB42BB1BE2B73D5D1434C95D"
02389     "D48DF1ACDFD7DD17C741A387F18E7D78B2A9A8942661678974AF1C8FED13D463B6BACDD81BC"
02390     "C2F3BFB17D2DFC0D43DD04FEFE39C801C8917D52BFE8D2E6F8C0C5159E0185805BF9DF9119"
02391     "4AC97F6448B9189028D50B274848B429AD4DE45194ABC458485617B59FC5B1F1606DB20283"
02392     "A73F41A6B36731F6F9E978B276B33B5DEA3BF87282D21F63787BEB1F6FCD3B21004E6FA793"
02393     "30A3345F2C01C8CD1FE661FF9AB5B484B3260F2332FDB00216DA42D2840D3180B6521B0128"
02394     "1E6D0F09664675531EED14A5C1794F3ED1FE634EF3D6E9D1B6F4871A45A83DB56FE2BBC52F"
02395     "CBC1E05FDCA884F98DEB436A9A238E086FB339E1A7AFE8E04A5110737F021C89EC5DCA1168"
02396     "E5CCAF92A04C950154F6209CDD01B287FC360B93E2A5661E7809FF1BDF3F71B40BA7564118"
02397     "94B340CBAFF9E0EDC8292E7C316ADF4F3B827F1B82A406B53AAD8D2D660FD0CD2D6A6C693A"
02398     "73511FBBB97612A8CD3F0E177EE38D46100E64C4C540C834DD87E062DA2C6C3BE8A0BBC46D"
02399     "2070FB8D9E1DC396423D2BD1DD22F1F4FEE0E24F48CAD9241990B5F48A50B3B178A9C1CA71"
02400     "F71321226E98D1F79DD7F3CB75E0B0C16915E34A92750BE9EA173C276421CD0E54319ADB0C"
02401     "4E5ACB4BD599842E5028F613067083522F0D2A891BFDB190855B9BF36A7F2EC81EA89AE11A"
02402     "37B6B89126DA7C51A4BD0C902D0DDA55A4286E36835D2EE95EB46AE7D6B919C05D42E543D9"
02403     "6D22CD241A8E6985A18F29184586E6A7C10D4A955D0A1983EEBEDC5A9FDE306FC366F77A25"
02404     "CFC10E5B8732D8F40D068224177288E9A13408F06AEC841C2E8612E6C794DE216A2B758C67"
02405     "6022D4C0B99A4A54166CA1153F43FF66AC03F183DC30E9C83E6EC3BA33F63F262F39838E5C"
02406     "044625B4CB1DBEB7ED957B8F7CCCA6C63B837ED93B2C260F276175F2F95B235D4D1DC0C901"
02407     "4192D72CFE014094F29A0AB8F0A9BE896067D74BE5489F7ABFBD7EF442579875CB17956699"
02408     "91438C0B6FF13E06D5BA805DD2D0DD06731ECA45FC8CB9D7CE568921A60E68E31F31392EB8"
02409     "2A39982BDFE8D953DE2CE0AEFFBF0243E43ED202FD6FF9E075A90684FED84A4334A8E70997"
02410     "9FD5B60BCD60D41907D1FC5D927F9CCAFF2DE85AE1350C8EF8D68EAE38EBD0DA0D252F0B5C"
02411     "82E1C3EF78CADF92CC78BD990B06461BD77B78635C406B4109FE58850D2A97A6C23CAA18B9"
02412     "AEA535D344E41E783B767A439DE0762A23D18244E76D3219A10100264EEFCA45F19022EFEE"
02413     "72B856FC34E59ADF6DFC78D31C8690CECFA45321AEE33DFD40943B158BCBEA764C3579FD5A"
02414     "06BE297D363994E6B959338A3B48F14A3C58CDAF1D91EA1711FB39CF78F04E72B293B80847"
02415     "516DA41D5024CEBFCBB399393E59FC4C89BA310377AF576105CDEB4F6DA314E990B1513BE9"
02416     "DF738AD35B1460DA23326D2FBD86722DEA5D37CE5E34C11408C01E1881BCDA250C7FCAEBA9"
02417     "DF85AA08FC9503546A55F2B4D6A7D0FAB2F50CA8667C7456DF6972273B62A0DAF38BBC7D38"
02418     "B5B62F2F1EA8476F300D0116598725C0903B0323779E02977F33AA8C45C2933B412490EC17"
02419     "FFC2AB2B4130CE8FED8D34419FD15971EEA5759EEAF9F3E8FB4FBB26615B4286AB4C2AC57A"
02420     "6E4BC80DDB396043251DBAE26ADB03767DC9EE4ED3E0FC7D74C58175BC1F393F0CE5D16589"
02421     "879C6D53645B861B079B298E056E4539B2E2E70920D3A273398B4E9247771B8A23B276DA9A"
02422     "06945DCF343E7A4C9E7B06960A1F624A75B1B48AF0",  // PRN 17
02423     "CB9F5B3A549646C4555D2FBDAF2A79F9921C54C6FEA316D7713D64FD790325A87A30DEDBDF"
02424     "0ADE3B7BEA52A0E319A57084A79DD706228989642E78BF51B7F1246BEF6F1B8724C57B0A36"
02425     "33FCF4440E9D57DAEEB343283D642710266386DED8FFCFD68B32949FAC691E66FE9B8D736E"
02426     "BCA6F4551B39E79557154E9AF590B6F5CA1C6A0ACA82FF4B839B6F9C555CC57255F96B2C96"
02427     "3536E48EC9B5BCBD270110EAF7C64AD9DAB53BC791CB111189117A34A74B4F23A8573A5988"
02428     "6BF974528D17B31CA9B6A6B7EF7F923B8915B640C599D2453941CD16633383F36F9849C016"
02429     "5D94698D793A58A7869AAC8B4C007CB9E43D8DD100E9504B1942CEABBA733A242599D5B8C6"
02430     "0AA42F4C8BE7F040877DA2648D083A2272A46A90D215D96DB2FF1ACA6CE24A2936C18B7A28"
02431     "3615E8E6B729602D018174276E2ADCBC869C43879B6BD0CF22CC0601F7491862971F65D3E0"
02432     "BB2474549FA18964B14D9EF81429F08383F77E8880E858B3F350076B709046284A4BF500A6"
02433     "C7D9D694F03ADC9753852285075197FE5F30F5FEA91C7740AF38CBF5B984ED852C99FD2711"
02434     "3F690CA7BD5CA46DCA330A7E30E7AA3AEE8CB62251DDDD62DDCC29C94D79759357FD928552"
02435     "79A8516ED0B07D8F2CF6509DEB6C180AFFE120B1C06A4DFB1CB76B9DF3113B08BE85A50BF7"
02436     "6480D2FE05E75A4F77909E0D146EDF0D751A1099EB2BD019525048BA145C7E0B3FEBE21A13"
02437     "A41A9BACED48E1A4ED32A57FA061243EE840ECA69191B1BCDC0358A86DDC98920C45F1FD19"
02438     "3D529C85B33E0135DCF2B81C7FEED32E07A2431755F3EF3EA685577A6884041C6600CCAEE0"
02439     "E4182CF54EEC1B078D74A18D8C81E0678A7D1309E494938083786C67436FE55FA955D3281F"
02440     "7F6C145B7C769D9019AE1B8666834B51AE95BEE2A8D606825C7609FD3F24057425B27D0157"
02441     "3FF353771BD47902A9BBC41B2016086A3F9E366F96DA8AE3C62123DF05113980D8FDEAC5B2"
02442     "C828D44C69F2D1420BF8E7913E59FCB88845A7FA6A84AA296875CA6D2FA21A4FDF8E75B5CD"
02443     "F0E1C0D84FEE983A469FEC80F02923695640FA0858E9B91812DD4D91AF0DFD97655C7FDC60"
02444     "EFEB19DB5AD7D7D0A5F93FC88A2AC1205D1BB60DB2F3BCE203C802752350B5E8240B6FEF4A"
02445     "38D8F35AE5868498DCCF1BA047DC56D67F15E3C3807BD5638A9494DDC268378DDED39A910A"
02446     "6C7A59E6738DC68D1A531C7EFA68BE06AD20BAC86BC36083E45AB0A314ADAAB741C6E24A53"
02447     "0B3E3FE318E4A1B3827057F944918664A96277528D185E821093D392012803ADB63A4DDB14"
02448     "983E844392542F2C5A978F1911A2720E201FBB862AE98C3CCDA720A244B5B58AE601C1ED81"
02449     "4B0753FCE5F51E7ED797AAA98FCC833EA272D86A238524C07349F5DD0BA4A2F6281B2EC2D2"
02450     "9EB546EF52F07B832053AA433A7E8BF81C6C9F562B603421787BCFF621A69ECA5CF07CE851"
02451     "2BEF945DC744AF403B9DF0FDD48AFFF264D4AAEDBD8702863B5A345A733328466B22FFEB20"
02452     "906543639F0F6F85F3F7DBBD28034019F4D517676C5D8C8CABD4118768B10158D83BFF569D"
02453     "3B71F30AB57B8337C50674E2290C950E9D48D6527E885A0F7CF580717CF467759CA0D41CD0"
```

```
02454     "3A91B75648DFCEC5C5BDCEFC2EB2007AEF2F01651A328DE030546933C1BF88F34AF14D0C1F"
02455     "0F37FC8FF4E015DA5D25D4A1D52EEC44DA8E832F01BB1D9FF6D067021D4703E8F2D511E73A"
02456     "DCE564EB53A048864D191F21C759F0B629E0053845D7049900924832DB7831CCB80B36644B"
02457     "F23EC71847286C959BF54C6CF9EF075B1EB7F43898",  // PRN 18
02458     "576592F533BF0DC0AC52CA27BF57894E407D8A264496F2196DEA6B4706CFDB4E108E07FD18"
02459     "F5B2253EC3BF1914BEA0E7B8EAC0EAD4C6A3D2E0162411811F8579262AB9F3411327C320D2"
02460     "7C7DAEC764EC5277EDFC2635ED39B0C497365B8C3633626E66D83DBFFD8E24BDBBF09A9706"
02461     "1D7F530F69D929DEB7AB12488951D753EFF9F8C8DA17F3F4B17ABB9A1955F40A4EDE9D058E"
02462     "6F48BF1B5533D91B55D333CE4AE379E124CFD375D82DD97B682D80704EFA3DDE799DC6F7D5"
02463     "E55B72A99DF2A89159CAC16B47F297A467358EC67A2EC72A01007C47D757DFE274893BB0E1"
02464     "41337105C46B159B08A0414A7EF8DE0806F90B5C5A9362E837191DA4573B473C6E4354E62C"
02465     "5667F75721370F036BD6CA025A49C15BF63A8B30AA5DA01CCF9746222F2113AF119DA4370D"
02466     "CE98ABEA322483C127CFC4AB990496CB53BA7D20255E63368FFDA40BB7022A1FC115B91ADE"
02467     "56B2711DDBBDDB9FB00C5FBE928F3121420288CC9001F865D44ABB4D78B3A1F2BF6B1B2C25"
02468     "1AB1075A65A233C4472A460FBC76ECF191313D700F73BAE29795EF9A5FBD0DDB5E2E04DD17"
02469     "2C85E15CB0A63EFC2A4E317C5FF01965FEB3BCC81CBADFB201C670F4C8AAEE2E7E75F1AC4E"
02470     "4AE0B826FA9F87B300188546E42F3F142AF19A4F91192741ACA71EBF29A6E1BB0D00A45340"
02471     "95A8B9885267F27DB9B676009631114A2EFF9B2D4C10111195F0C178D2634EDE70B4C0C876"
02472     "13CC9344BF2F62E9BCB42C4CECE15AA61A0C7915E51A5645609620695B4D0C913337DEEFD9"
02473     "195CC9545A8B1AFEBFE5F21BBDFA2F140A4D146ACC25B8332861C2EA55BA75086FD948BD27"
02474     "923BD2590C62389CFBE89498C82279258D11689613C21CA8F1788C4926EADE381C9CDB41D0"
02475     "9563890433AC4BF9421475EE42AE5735D1CD10653713BD62299F66F793FA095B2E04EEA1CF"
02476     "59EFA731143D5849032DA833E1660A95117A2ED97DBF4D22FC0BEC0B5CB8EC32B63BA18B63"
02477     "F737F3ED775B6D91E052FCDBC0C33A6F4498082542B039A26BCB25CFB1B23088D9BAB9DB12"
02478     "CD18BF2AE73C78792BA9096AF766114337FBC06CD63BA8D1C0A5E5C6BBF5AEAC38D6B4EC29"
02479     "22B8ABC930F58CFAF2C65C43422193F4AEB436C05AE3213CD19F58CC7623886525FDB77DAF"
02480     "08CEDBD5D48233D63F1D6BDA7DAD15E817775DB4C9DFE754F4AEB2A94DE6F883961DF23801"
02481     "A645906BEEF9128F3F6C9FDB57E6D7B5EB46326C4C39F2C0694230CC28852D54281417C4D2"
02482     "4470B9EBB625AD1D144B4889D23EC975CAB65A855CDA704128A61CC7C50C5ACFD6355570E0"
02483     "7CE55578E4453CEF4582C368E585CE0F08BF786CDC2C0158E1A10971AB769872C52DAEA5D7"
02484     "C8947A2B6300815B42125A7B8773E6E8C79239D32D7AF37CB9D4E8688CEDCD71A213795B0C"
02485     "1BD5ADB0AA337989ED3E4E65DAA164C26946978C9468EAF74617C47EE4328C51F41418F2D3"
02486     "5BD33D47A9A1ADBB7C4F5A968946A7A900C293C65648C3127EF62D58BA9995B19B10D931C0"
02487     "7D67164B2E685FB840A019EF4752602DC9317714D89F6D9318256EDBAF002C40D5721525AC"
02488     "0D2BD2B4FC64138F0FCC5EE648E84AB6908C1AF71408D51D31F7587F1BC8E19D1C045D8921"
02489     "B447CFA9287AD03B4EDFC3FB85C7792CB2C479BD5A14B72EB8D3077CC511B0A7364C80C32F"
02490     "11CD1EDB4B3A0806BBE9D7765B091E04F46620C23DB801A46C544A935FEE00F0A07927171D"
02491     "37D3FDAD3DFA81C40CE60F93EB65E667FE032A79DB745EBCD7EDE57FD51B647CE256244441"
02492     "99EE2D9647FAAF9F43554433B25CE78098BB33CF24",  // PRN 19
02493     "A888113D9682BAFE58B278C036FC2EF41CF8C156C39D67C378E4E0EEBE750656D67250F844"
02494     "7C420E6EAEF139A11EC90B1B50CB582E5089694A4BFD843602382C022085141A7B8AA1DC5F"
02495     "68469A2C2AB04BBE91EBEDAFA2A19CF0538AE9D47421C3F98084073F340F0292B894645225"
02496     "1630987DDA49E56A79F1562C1EF57134F036A5DE932B2AF9ED5D24D28463DA55F4E7C29045"
02497     "DE9CFFC4372F7A6AEBDCBF35DFF2D952D90BD4650FDB3BD68F25B65A7876B65C5AFFE8B9CB"
02498     "06CF91B88A588136C80D9CD361AD83F25B6D1E17D161817E64CB06262AFED0417DF26B9B3C"
02499     "0AAB0A06B1310A7375352C46A410E167366AD768559A6DC618B416395F432C0CAAA8D10F88"
02500     "E9765F36B196B37EEF1F1C13C4AC08150C9E4CE45D9B1663FA317263672875E0BB5CFFAE2A"
02501     "132A6B431AA7099E47E7FC8349D04EFF0B84C3E8A743572C4B1EF38F760656E49FB1A4DF4E"
02502     "2D1F6B24090EC675D299B32FC5EBDE40BBC2B1DD44CC01D1935F1FC084749DF9713DE41365"
02503     "D8CC9AE5041D60967B25E16FF1941A9A7336DC2D46DF5AB88C14A4BDEF077EACFB1F08B834"
02504     "AAECFC75B5E3C1BBAFD3EBC73E187D862411AB97D694B2E557C47E5093FC15780805518758"
02505     "B4C5B548306F92B559DC58475464C6DCC18D2B3905FE5F60B4530F87716107A1A4EB40090C"
02506     "23527895582B02141386A726FFA11E9930B6099B22816DEEA2626E60F4B17C5E175A852A2E"
02507     "174D59FE8219C190D0D6C50C14A3C5215BB193FA56209CA4D14AB2DB5CEAC2B3B20CEE6DA1"
02508     "F798B681D2DADE8DD13674B9156E5C3C4AA09B1932CF69D62621DEEF990AD264996069B01E"
02509     "6F7A21F2E5996251D09FD955CCA7DAA902FA7D46E03AABD4FFC5A1A873711FBDFDEA0935C3"
02510     "99A0178CF2516FD03746B86B5A80576A34119373EA7F059A0EDA3F0D18CF33ED63DC27C815"
02511     "7E0C43C537E86FD37E58EB5ABEB5524FE8E19B2298B9DF539C7C31FF0F7AE96A56FD3806B8"
02512     "1088DC13224131998DF42066802ADC2A08AF6F69D36A7B9B518430C5532D3AEFB20BE0BD1B"
02513     "F4B2C968AA78331C614EE51B8E677B19A98E30756B39BB6A1AEF0E91915BC91D98E86FFE75"
02514     "84BA6ED8D6D51A132EBC124C056FBB9363B8610EB3D7A77A3E581C02E7D25184ECE47A688E"
02515     "718922E4672CA9567BBA9BAEC81CD8DE7FBB8CA04D20B51C850DA2A4AA63E87FFB27AC89D1"
02516     "8FAC40AAFB21331AEDABFC3545333E57C73CB7D6B601B7180E4489269001A6786415F4D1AC"
02517     "D8EBC3CCD8E78F6577A9E107ECBA3EF5EFA242F80202C1EDAD26D9FDB8F43F70D250D899EA"
02518     "AC973042149B8F0BF7327D72A49222B5C403645BFDF3FBAF78F2AAC71B9FB582346CA456EC"
02519     "183867409C18E11A1BBFE6FF5B681B0F893778C87A9BE8A67E196919945F4F22D70FF5D85C"
02520     "55D23893E7EFD9D685C41B28B118EFC905C450A9E91112C5CEA2475EBD07DBEF0A125C298F"
02521     "467377E2BF8D4BBCCA25A2DCC4AC83AF877E1FC820D09F31888802B3226FB0129AB9E47FA6"
02522     "5BAC527CE74BADAF59F95CAFCB6EFBDAC889EE8075CF7C71518535DA63E7BC9DC566841712"
02523     "3CF11DAB1070B050DEA01F46BD86E59C5B3175EC1C0C1244E0E7B5B6FB5D26979393E430C7"
02524     "B79893E3BF819380D206438174038764B7196DC0FD94D4A9CB8382CEED4490C43C19061D2B"
02525     "ECABCDF386F1512EF3D635601E8E9CDECE9A452A0E6C6AE0D6BBBF3AFE226A46AB3285531C"
02526     "7AA9E0FCFDE0D9A557F20E9802139A7C8EC2B2872CBA3AAE87F0BC6DA226BE4AA5681B114E"
02527     "080A399BF7480E8583DF2505AEEE32E45C4112FC48",  // PRN 20
02528     "DD364986DCF106EED4C8705C1A9FDDF9480D337BE5C61AA3F661C6968AA01E3E454698FFAD"
02529     "F366C73599B987DE85005C9743A2416E637E4BBB67FFC4D35F26B7539C33256EDBF680939C"
02530     "C8986F5BF32D2ECE4905A4F6A1727BE1131771954856A1C66A402FDD0DB86B51E9C02E6BB1"
02531     "36465FC4253C8D98FED1582EF8AD22828B6A35580FD58A0038A9D97FE2F661F31265804183"
02532     "9969752C4ED321D4EC8229C5753F399029B0BD5F065D5A19CB4548C86CCC1F3A9873FFB9E4"
02533     "F421D033E4EEDD4ADBAF302E24371EA97D817AA33C72708C1EDAECBDE7C06AE0C95D823751"
02534     "7666D0FB43B2434BC50EC2BC8DF1E3CB46099BAEB869B251E79E9B175B75FF62F408A5FCAA"
02535     "4F6DEC90F97B926272389696B5E38D26B946E0575907C358DB17D046D58D061683E07F1B07"
02536     "5C10934FC4260043AAC28928DC877D27DB9BAE618910564EC3EDE3C402E86BC3981203CB7D"
02537     "CB9940C37D948556A57A11488D1332B05F4E5757D470A22D0B86C88DC456CE014B673CF5FE"
02538     "ECAAF26F4662C9B243F46969025692E445ABF375AECBAE50EC9E48AD25EE1D1D0C16624CC5"
02539     "BCD1BEA8C8FFE3579C50B732625A381002115A1D69DAFF5844731F41AFCDE42CCF1A13FC98"
02540     "A64FB9769F03CA631B1C3ADA6A98F40343BFEDCD80966652292B60DE83385FFEA9CBCBFF51"
```

```
02541     "450C768EC8C4FF6FDF2F76FDA7D5886148168CE2BF693BE0179DE250D41DAA80553488D4DD"
02542     "B01FF97329E94C7DB33005B498FA8408E13FD5BBFE079F6E40EC5C8B735EBF9914B932A824"
02543     "41D9D5136C662294956CB52939F4F7E335A3B9749CC1B9F374AAC89C9AD1E718BC469CA21D"
02544     "2CA467F5DED54C402BA0CEA000DCC878822EE3E1D329CE76E421305178FDB31F5DAC177838"
02545     "197C5B46A4F96721139695EBBCF39F04BC1BDBEAB567E8FD812C355CF72785CDB3A6909F80"
02546     "1BD2F448B3EA0C7F96D09DBE4EFFB1CAAF2A47FE6BB830FDB937CC5ED2B485AA6B85DA799C"
02547     "1266562DF181005A345C62BF94004C1B8D85AC06D114C8F2C5E2D569C709251EB929237C76"
02548     "E9CCFB9CB39206BE3C391FCF5E814E42AC2BE1576B9F0664B20F03BDA1F6431D1D1CD4C078"
02549     "3CDAA2A7AD32281A06B2BEC53EF1568840277EA4341F5570388045526AB214B477EA23023C"
02550     "13F4327DD0C84CD1744677091B21D8F95F38A8904EC5D6542AD6E3A68B355D688B1F6CA50D"
02551     "B7181272B3D564B1DE41C14575217FEE44E5B631CA49995EC2CA787EEC4533657FFDDE7370"
02552     "329C9BCC301B8785F7A307F1D06F88F21F5BD19F94827C128F799A9C354A021146985256B3"
02553     "D3E8BE46AF6E370EE193BDA3494E0FCC7AA287C3A5B93B638AAF9BAEA46288DDF690A46401"
02554     "7765FC2F769A5A930FD3E5A5F399B2F972EC286E81062B6BA85FC04DBBBC05B9971799225E"
02555     "2F0867B55CBCDCDD3B85D6D56C6948B03E91080FBCE3B2C5438491050BD2E81C41D5C2F679"
02556     "0AB1F327D2C7023FD891F8D6E13D6F69E7AB1604692F21ECC7794F028A8306657D72B58583"
02557     "01F5296A5FFDCBEC4863D0503749E9B483E4760AFC05682304EB17D67E016553FE9108895D"
02558     "F3EA98890172D795009AB414E4539C47200C56828919DBC1AFFE040FB3319A068264607CE5"
02559     "1D387B9291B7483CE1C38C1F059AA86B68BC7F852D21BF759A56557AF5B4E8CAD60356AE1D"
02560     "9D11AB49CFB9F627DFB1AF5BF8A6D85A775B66767616FEFEF6488C383F5ABAD4E06DF0B6C8"
02561     "2BD2582B284BC1D4B27168525C69FFE8117FCC38FE758B2DF09E38DCB1688789B2AF3A0501"
02562     "42E50787EBF2098C229E6965FE134593AE47AF7FC4",  // PRN 21
02563     "B59F424CE3388B73A01C7279251C7854C664803D834EBE840CF3851B922071026DC17B967F"
02564     "06E8467AD73F8F70440CF20178BD31F8CC315AD8B3D213029D08746BA9FC70AAFFAFE2F393"
02565     "D0339BBF28FC15B8581667E9B43E0A1534AB70BDE8954B30AFCD4BF10B91B8BE2AEDD6489D"
02566     "590E0947B6155A1E0B4E7DBB7D5DF08A340FE33DEF34330EC77E65B57FEBD4745B0A07CB8B"
02567     "CEF0ABA92E387E2C2F80230818DB1F70F65F872A2C5580A6C39B9EF600C797FF0C61641100"
02568     "903B5243D847C7243EADCB69F69D75F4B1D921C72A4B5B67D11B78F84BD9682963E075C763"
02569     "50BC301641E76BFAA23EFF0FDA33B1A41729AF935C09C1E5257F84A4AA0AA3A54525BDF1E3"
02570     "A51EDDA6A35A2C2B6DA82E5C7FC989475D99585A9DCB66325B98C20BD7550871BF66979BA3"
02571     "D58562C1125EFF230FDC77A3BE8A7A8CAEE2F83C62688E0EE5D72D4C858624C7362A57B7C0"
02572     "065670DD8776D1E97F67CDAE4110E53E78BC17793CD34F11089AF317407B5F43E26424A752"
02573     "DA5AD58470FB8FDC193D2B28FD4620C879BD22A24DE4EFCE4FC46D7A5A4350136F43D21698"
02574     "AEF1B4C318156D574E93BDEAC252675BB1F1819B651D08E637538DC14505779521CBA1C569"
02575     "5CE7C5F9BFA3A7E68E14ECDE399144E6D514CBEA519FFD2C677310214DF222CD79C9B9F763"
02576     "060C48D8FC4E02DBCF0D6396F6A28BED4ADB4F69FA44BF6F52E7516E77CAA981712F10E94D"
02577     "AE683DF1CD858492A6B7C1FDC92B3A8A4B121B1C8B8791E18062548BDD31DC544FD94DA7BD"
02578     "77A0BD18B3940760A3746ECAEFEDBB6B99562E836421F24C2F2A99F686DCF9C57E91074803"
02579     "1521E359D21A69933D3B3173D520E7B8810035140A9F30DECF99A117B8C81990473378C58D"
02580     "1313E29A9824034B86D46BDF15479E8273BBFC4710958348B17C8EB8EBF8581AA9D1032AD5"
02581     "91B7AB62B3E2DE12F36BF0ED71CB0DB342D5233243F466291DAFFD75971427F37549BC74CC"
02582     "8EA2C9D61C55FF4EEF46EC74F37E30AE9227F9A346A5A379A386E4706FC6DADCC80FC7CAAF"
02583     "6164673DF6B7505C1CF8C1A2C262F13A65CF1DA6D3CD277494735E5E0B4366804A816373AA"
02584     "0A33F3FB3566D9811D26380EF7E049F004E5540E3A984793553B866E364BF3ADE51ADEF86D"
02585     "37A9392B6D0D518C0F07F80E9BCF795C540BCF8BE8AE72C0A0A67160F296E30A0F5727391F"
02586     "82DF95DBF2376B82EE6DD6C87C05CA191AB8F32671C4E404734D6626DA299682D9A49788A7"
02587     "D961B737282CF16DBCB233124C5EF4800E1F25316763BDD92F090088218C071E91EF2D76D8"
02588     "0BD05D23C6C06EE975279B0505CA18623A33A4D94EA882A6C0C5510986D537B49BAE93E363"
02589     "833A8571F91C47F022A6E05FC7673149128E3876AB6F423CA96A7F0F4D5AC1682DFC05BD4D"
02590     "3ED478218D40A42A894DAD908A587A83299D9CA685C4E25155760F53B27EC3F3E722E46E63"
02591     "BA2CA0D353C60DBF985F3E10893A0BC01539FBE6126215B090F1355D211E833BC019C7BB35"
02592     "1B1A51E5286A178F70DB39D3A8DB65E510A0A93FD717FF30BB586B8FF49D62ABD56F0AA4B0"
02593     "1D2A8E6A3BFEA54FF1D36C8ABA385D0974C7F29B25BF12B1036E4926F0E70E6CDC93233E21"
02594     "1693F7B315C15594790BFF888ECE21352C8FDF906DEB1085408C877A98BC5B145BB1D0945E"
02595     "33409EB01E243D386492BD62847AA67D4EAEF5E8B84468F70457470C1FA15BBB44AE4D77BC"
02596     "401BC4128654B52B2B8CA48721DBB267EFA7F0B2ADAEF8C2BE9516A42BD2FC2C7AFB8D0D71"
02597     "FBB8A6D7034892078F4FE50A3B63105B31451A7910",  // PRN 22
02598     "FF81F6EFB1CF3169B63C2F2544105BE42B20467E1FF2E4204601A486928A77571247AD40EF"
02599     "058003E423521B4196118088D773BD1694CC83C02219F5E16C31BBC7350A023E4FE0720DCA"
02600     "713444B455DA085653B539A555F37AEE9357BF538E3EA6A2C3B2BD30EFC32665B597A611EE"
02601     "9DDBDA9A9857FFA648CBB75677DAB55EE8528DB18C4A188C693A9DA0D7DD736401DA333DDA"
02602     "13F572D4652D23F9016CE9D39605B5E2F341F487980331D88511836C7560B9F3E8DF6FF4CD"
02603     "B7F74459BA199E081FA49BA205B368B4D519B2D0ABF92405409EDF1DF280A04F398758E551"
02604     "1B6E6325CE00C57186A0938974207E0059B2B8BC81156E2FC0A76E84C5BBADF4FDFABA379E"
02605     "6F960DFF5DB82FA306535F0A8C4D925246DF1D9B8C7A711554E9087A5FDEDD656CAE74357F"
02606     "C6AAE8EA0C72B74BDA7BD4898C6FC7BC50367FEDE6D5261B0B72E171043C89C8C2149D0146"
02607     "6B226F6A1F3F3FF378E34A3956C660F0FC8022E15EA68DE5DAC57BC370B30F9DDCA6404C5A"
02608     "152831230DA9EC3662DE701767953AE8DDB9918598844202DC2AB41A0ED711D640F3C60A8C"
02609     "C69BD8D9340A32330F9D27E7A023CC99D173BFE6CA4F0604FF6E5B4C59EC2E1BD4FAEB3321"
02610     "F755B2494343BB6E647210041A1B96A02B7755EFB6B482B3C7122EEF9242FD64D0D7BE24D3"
02611     "20FD33ED0D4DF2756BDB13BBFB622E7E928676D228DC395DC32465E0E3480B7ED2EDDE0410"
02612     "1E9C0F8020D48F3D43A0F756622A3DD866B966B7B0BFC1F40B1D460D6A54A45ECCF5AFC93D"
02613     "AF8541FCC62A5FDBE907D2DA6397B3F7B535DD8E30D400610971E49960859DFBD4A441F2090"
02614     "21E9E4B05D3EA0C80A738F0C4EE9B03492692C543968122FE39BE1F2E5D682C1FB8549B12D"
02615     "6F32AD5938FAF63C5F490EAFCDA9D7DB70681B5650081B098D0BEC87013C70F890666AE984"
02616     "C5FD92F6939419AB0C1E981FF6EB03B1A48D708438A8F6AF81515D8C42E0DA5E13DC07D22A"
02617     "0CE018105BA075A696BAB7CAF420A65E38EA1961B61ECA43AA0085BF2BA6244EDCB56EAC6D"
02618     "ED736712045911D7724E9DDED1C05DC7205580FDF52143C07960AFAC71528C1B87BFD9D21E"
02619     "A5DA5F9649AB037CEFA5713647DBA70A2D0456A24AFB0B55B8C859F9C9617E83E12C0D9633"
02620     "97DDCA09CB16E22A47EAD91206E8F3D925444AB85D76DE73A43AA45BD3B0412FBDF254CF27"
02621     "3E5F20F6BD0D938E30ECDF159120BFB28FEFC0FEA1DB898D5D1E674452DA86C9AD4AAAAFA0"
02622     "7AC6A304CEAADA8AC82870313BBD56E746E7C9C183EF5B53BFB5A374B955D053D1CD7E8EE2"
02623     "5108BBB8F2A60663375832213D1B9064C9F12CA77835ADFE28D1A5E1E65FDBA45147F05294"
02624     "8A507180EF21052D5FBAC2B3F4E46DBC4ECC7DE08D6FB3B6162432B25EEDA07911CDF4E98A"
02625     "219603E19FCAA8142621F546C898181E1761AE7907F624A5D01E36612F06ACAA157BBC316E"
02626     "9AA7568E359803A408B0C36B4593DA1B4D4D6D1E6AC290C90D19635394E10F0BE0B5116577"
02627     "7517AF168F38B702DB2BB43ECB3DA55D9CF9E8166D99B971137F6845653B8CFBFADB778E16"
```

```
02628     "A8F64BEF8DBFC6DD4F6F9093FE629C99585D59445ABC30DCAD609903A26C181FDF48164204"
02629     "3066E5B0561F13934247D19975F136C0E0F4CD41ED3F8E22739FE2D88358CAED503D7C36FE"
02630     "755ED34B9EF871F2323AB593A0AA8BFDA8DD3D86304E206E5B4FAA6B905751EAAB46FB576C"
02631     "3AAC9387E38A02A6AF8EC72BC953C4831FED1337EA65241E4D94F90A0E99018DA735E1CD4C"
02632     "835AD9F301A618395C8FD4C982CB5282256F8A4894",  // PRN 23
02633     "CE812897E0C0F7BA13D73227DA8D57F576A25F233384DAC16B5F3AA455222CF8B1C3F0C69F"
02634     "6E1EA86C6BE0DF818820D7611EC41F6078EEA4461C336D02847EC3C5644B85EBA26B68C592"
02635     "7D67E36BD381F8100B6BA90852D529419EBC8118A5354706D2CC769C2402E517A86809687B"
02636     "A7E0B1940A88F0B2586484EB7D1B63E119D489A41691D0C1A88543D43D8962D17FCB8E91A6"
02637     "CBE2635F4D0814A8C9A4DCD7767CE7D62E8BF17117A6650E3E8933DABD4CCBA43F40ECDC46"
02638     "EFC541CAEF6271E87BE92E63DFB7C2E5B7D7D2ABCD56A0A19E3C9F75894860A850029ADA36"
02639     "F990DAAAAC5AF56B157475EB18943D7CCB8D99CD14813A27036ED1F4DE3234BE28FBA41304"
02640     "6A26113124B5FEA24F4074E991E163AC879520669ABE6B1A0203FAD6E4082CD5ECE201A86E"
02641     "C87676E350021FB2A1E04AE9B19A03152F8244542B4D21B4173D05F159EDCE12BA12A57673"
02642     "63794DEF4D390F56208EAA7680AB527C2035B065DD58AC8182BD5660DF75511EE1E0F407F4"
02643     "EA72AB3FCB028C82D8FA3003B034EFF4DC1C179029C452BC4B4D75AD02CE9E393C15872F8D"
02644     "130AFFAAE9A95640EC68771EBE628C42DA59AE4AC2E9156534430D00164E3D2D28665BD2DE"
02645     "6AC1482EE3475F7E00F41078B672835B435711C560166CB4DF17B56884C8A66DCE5D4B42AE"
02646     "675335EF457AC20E4BE7FC32FEAAF7EF4CAD795ABA7F9A20C6703E7DADA30B63CD6CC21411"
02647     "4242C81A9614712271CC05867595D9938ED557007713F36D1A44AD2285A564C9B242AF6BE9"
02648     "979F7C474385680FD574D9D33B8E2DBC1318EA71977FF83E8977B93BE3EAB2401A136E021D"
02649     "A182B1AE68007FB36067507FF6900AFC4743A9A8503B7A2CEE04A1BDBE9D661691E5E8C6E5"
02650     "568D16F70C15D156F45DC41199A9C67FAAA36D8CF67672717ED0124E996CE855392F5A8BD2"
02651     "4EBCD644AE0CD5B1AD2711200130AA4F8D3B4599AE660A56EB6E192709CFB8890510CFB019"
02652     "5D7FE794B9EDBCCB25ABE6F3B6AAECF3FCAB9ADD6D270D3D5B934C46C8D6800CFCF99617AA"
02653     "B8192C28AB627EB0D6234F7C68F0F4241662A9F062B361FE89E92BF9254F823E4164AE5466"
02654     "1C2991A56C4AF8884DEDCB20EAA02D3097F2D5BC45FAD616F2DE3F37FE2DA9CAC838ECB3BF"
02655     "649AB6CCC386751263815 61D546B37C7432412974764971 84A861ED1FCC75BD508915AF397"
02656     "52866B39F1BF00799F3B254C6710D15F5F79E29766E3EB75B93CEEAD0F3D454A3A0A4B9B88"
02657     "534AA91B604424CA1F9801F7F16E0A5489CEA18CCF76537CE33481328C83F3205B59AB1BAC"
02658     "1467C56A28DA696DF2B08CD380049C5E0FABC6AC8C5C50F9481161C1D60B0976FF459B2C11"
02659     "679DF8239AF63C1379997A8CDD47954DA1A9ED1606A3D1D2700A2B9B0C9FEED839E011D527"
02660     "52CE543ED224E5241340CFE1219A1A0117E033F48B262A6C83FB13BAEBF036C2ADD4ACC5D1"
02661     "CD9C5C334F4EE7D964851D9A5B7CF5A38B3C1CCD6DFDB5D1AE3FEAEDFEB72A25C7CA779DC4"
02662     "602FA0E6D8C4B35900F634130D48BDB8291B7BA751256EF38FD6FF0D5662C1691B82571444"
02663     "2F5B633B30514058C778C78BBD6A70A2096CC9A375AA99CE11E8FE4388ED8D403EFDA3E604"
02664     "EA77820D4D7A1FE7C51B73F286520D56DF828B659FC636DE3DEC09FFBEC46FC143D943A096"
02665     "34A91194D77F0A70DAED26ABD3F6B51457EEC61F9AB544F544B21A1284F0E70D75DAED0B9D"
02666     "0FD864CC73236CB64EFC7C696882B1B408B6B02F35EFC4B5DCAAFC600330C59B1338DC593B"
02667     "C22C86A1C4F16D72E7BD622A3561F60362F3612580",  // PRN 24
02668     "DCD55CF8406A7CC9BA0A5224797D74777689C4FBC0D202DEA8B9773A0F20DB0438BEFD40A9"
02669     "96EB2E3F49CBE2E475DEEF73F112FF742269A701AB43D8E47EF94DA4056086D86B1FAD636D"
02670     "89929C60238133A5B61DC4B7F8DFC765CF0C8C7206F3E0DD3DA13407CA75EA20F172AC38C4"
02671     "2ACADC946B3B18BBB30DF871905777CFE0D3BE5363574D2545D0214D6A43E2D34B50D7FF7D"
02672     "9F2DAE7053736B71FAAECFC2C355C1DA645A995981E1D617BDDD14EB9AB230D10085DA1AE5"
02673     "5D1C9E55C225997D128CC2393D6E5BE9F1BDE1543DCAE918B38AFD5FAE6031E105BA12EA41"
02674     "62F7E24572688C54291A61635C9DA26BB367348473D4DD4AEF9A5C4E8A6512AB149D62F625"
02675     "D8D45576FAA4CE28E5C3D52D293A0D5C48821E1F5CDAD7257C97B3C1A6DD83A4ABB55F780D"
02676     "09964D4B3940D36B6709D00094B89F319E1C2CBBB3EC33A6F0398880B8079E6A39DE697D76"
02677     "803B04C230EE980765604929D60733BBB6382B4CA4B13AD3FC48AC9F944B97133C5AE6C3B9"
02678     "D295A760D786DDBFF346C873C2DCF3D2591926116A318F63835783A33A17C452AA6C620284"
02679     "BA99DEAF023972082AD2B370CF3016F0739103A518B25AE8D59BB017C8199ECCAF3E908F9F"
02680     "144C778E0F5F47FA06FE8A22E3B2CE3493EAA8745478D653F8D3711F832D9212D060A7F0B0"
02681     "B62AB9A3C9BF012B61A9B5E6DBAF4B48D90312504196 27E96076925AF95E2B238E8CD227A0"
02682     "F3D8C9C3BC68D5990952D6CB62803B8D193FB06120BCA9B195E92B995B86B92C455A6145D3"
02683     "F77898E457E1B938183E1D76FF61C163057A4D70697507BE020CC28BE6527CE306A5D2506B"
02684     "86152B5E3B954D5A660C79CDAABB2EEF0D135C0D4AACC7FC3EB5A3575D95070B168FCEF4F9"
02685     "B36FC3D55B84DDBEF10DF6BA659F70CB2AB82FA001D51ED0F040F9DAD4129C324BE8B44316"
02686     "C0E3E15B27C2A8D54A948585834C2DBD40337B61F58F3AB99B3303D56EC1D30ED82F29CF6A"
02687     "68208B07DA120AC4073F102F4D796E38628D3443845518109 98F6211A082BA225A1B7D63C9"
02688     "A3E8703DD945BE1D979677FEBE715396988A873B88EBCB10209E4DAFF939776AD14C5DC90B"
02689     "BE9C1A57581A145BE0D4BB47CF50721EFECCB50C96E824DF93D675B1A1B4D920CB1EE82B03"
02690     "69094DD84FAB4A3FA94F152F87525ED6D17A800EDB1BAA6C13A2EDB5C9B7C5A82AE5824AE9"
02691     "BA63156EDF78517AE2B357C1A1E8EF6CB12DA415A914FB69FA633527B8A1659292D079B580"
02692     "DEC210231BCA77465BC6A9516F2917226957DDDD9787313FA599BC2BDD14DEC9DCAF0F2520"
02693     "318A85C55DF40881FF65E4AC21F2C9EA3CC06651309E1F89982D45FD6EB5BD588B647A80D2"
02694     "7433EF85080A7AB2FFDBFC7D2B54B215E7D27A333D4942CD23D53142F04976093CC0344C19"
02695     "F9CB3A5A4D522E26B7E2C1C85C1ABAE1B4A360945A46D097A1F3676A80E3BB9CBED0BD363C"
02696     "B70C2B250EA053B183CFD1EDF2F6A211CE618F70EA1AB535E96CDD0B2FAB7CB7F0396B6DEB"
02697     "59745C1BDBCD452FFD70C95E45B88325A82CF07EFEAD7176FAF04C8C3E0CC4458540BED99A"
02698     "BB9F52AFBE48E42C636DD076A37C07C8E4EB6830FAC011936EC047F6D5B5857AAF90F80B5D"
02699     "33B502A116828BD27E30B3A69F351E22D943A482BC6BC41BE7BF941F3AA7BF85A2BA5C38FC"
02700     "930680FDE5A4E174AEC4156A755E7650CE488620D6CC62BF9D085FE1782EADB11541A5C19C"
02701     "9061734513316B9CA608A6E48D333A110B57BA5128E9331E6266174B85F7B7D1DB264361B1"
02702     "C6C044A8274A7C8DB19C253BF8DAB819E85130E920",  // PRN 25
02703     "79E4509D63EC15D4930F17DB3EF8E9ADCDE82C9F2308A0C9F77359575876D90A9CE5ABBBFF"
02704     "E09B8E145F35C7E91B141F66F74B3BA923D9ECF863A8A455086DAADC247872C953476A8577"
02705     "2079783A6F37C964AB1788479175F7B4E9651C7DD5EDA94F8BCFA57B3542788FE9547E0A44"
02706     "4D249976333F99710930CED2E93B1526A02BFCCCE416F0CFAA631B878BF23E41F95770315F"
02707     "1E169EF22B9DAF0E2F269E06B03B125CC6F20D2AA9A13B98CE23FE3A7FBC70CD05496ABD89"
02708     "ABA5BF58FBD8D88EB9653963F0452CA43CB964F7B145020D637905E8D26E8CCCE4EF892BC4"
02709     "8381303161644FCAA1DB9C4775E8F0A6367F1A842C3101C9F94730F6724257F2EE1657CC93"
02710     "950C9C1F20832DCC11E0001AB347781F21DDCD4786234C908772B553D7BDF4795FE69D7D55"
02711     "E950D4C6E2A76C44995CCA5C9CD539ED235CC55DF6A004C1BCA852C2813DD17F333CF67711"
02712     "EB8A5DA439D73271272D0D62B34DD457C7412A26FE2F4F23E5F5AE8C2F91DDF86B7048A408"
02713     "72AB7C728B20DF7AE2A0E8E52B00899C565E68390DFC83344DBEB7257CF8031664FD93DB65"
02714     "738013BA234035F11B611DEA4F74B1000724B2682EDA9639C11F4F4580683DDCA2D9159F63"
```

```
02715     "7976293FDF76AE2AC7EB5203766009577DF87D9DB654ACD64229495A5B4A94AB3AD8237B9C"
02716     "FA2B5AB6B7239A5146C777C4A3CCA9E1573D96A7C1B333F11D284AEB55DBABE6B4F12A421F"
02717     "F7BD442AB6B8A7FC33A2087886328B33037EA78D9A7015ADA2C8F463AC1FEA8FF3EA886823"
02718     "063992176C31152573C10516BE386CDF81A6568D8E3BA2FF195B23903C95B16277BD60D238"
02719     "F309A7DD6BDB4ACF4541A56DF5D3D4BA4078E58641C0453B1BC8DAAF6F1DB28836D6FBA347"
02720     "4D69870E07997DF4541BDA5C62291A3360F5C5840752A25CFE366186F7E4E21DA0704266B5"
02721     "C14ADC1541E463917D5DA677945574F800BF47FA028E82200F9E53C48C887355D9E3CE5FB7"
02722     "78341A356423D00B18D3048C92A80279C2785AECAE9EA57F828766D6F953AACF93FBE8A4C3"
02723     "2C7FC9843F466141DD018CBD05ED8A16E27B43C5C0EE16045D39CDCC7E4A27EF636408E968"
02724     "B4C99C895CE77288C819701EE7184DFCE889328E4CD2D8BA7BE5A1DAE50E36D562BF2441B5"
02725     "6242EF9C4EE63022F88F0F77E6094AD7DA93F672C06B6BF4ED5523D1DBE7D8F13892BF90DF"
02726     "FD5D2A3BD0E9B56EEF03E560BB28650D873116C0CDD00148C90CFCECB6BE9C4BDD7603B5CE"
02727     "9FC458431C00B1D704702B552D847D0C89265D842FF6599A150854848B730AB33CCF2BA653"
02728     "3638860BA564540C6678061CD97BE3D00B103314894AF13999FCBDECAFB5A895671DD1ECDD"
02729     "95C0C94A250B014CBB657ABE3028C8D6CE436FAD8FB7BDD12AA2DA32FA6435889F1B8C16EA"
02730     "8488FD6AFDC7AFBF5358A31B23E0C05E315B60F180986BA17E26621D92CB639A0B815E65D1"
02731     "40BA8A68F99BD42A2F428345AD3037274E515684EEEF018BA41A7B0138D7FF2C0B83882319"
02732     "2FDEED577028052EE73D65E92812E4C33FA3CDC11234F47F702AD81F3829ACF1910B9D0016"
02733     "7A3541372DB79996FC8F4B15F2A56B272B7DCBC4B074A36543B12B3A5396B0635068B62F81"
02734     "A56E33B40516A376D4ECF7BEA4F8ADBEAEACBDBE23F5F311A39D90D0B11BCAD30609F6CB4A"
02735     "657CC620CF8F727E30F77618E5870F72EDA78EB89399236F7F5432560049262E29A2979F83"
02736     "2F2B57CEB25BBC2405A3098CBCD3CA0CDF631B37440A47D316F4860597AF78E5ACEA13E8B8"
02737     "B5420210D7BF54B1F07A433E7A2B193D7EEFDB2288",  // PRN 26
02738     "B63460020483CDBB47DB98D8D720FAD57072D41EDB5FF2A2CF65DFB70ABF3524003842F7BC"
02739     "F9D36F8E187F86B864C5FE0181E5E3D33AE11A50E67F56DE93387F5CA7D1641CED92F7195F"
02740     "2ABCC7B32C6C7BD8AE561E36A6359BA6171726A2F9B00C7655C7E346310C6EBE10D2D470BD"
02741     "5BB7B4C88378DBE2352AA45EDD728F42A3DF3B5E80BB6B6BF55CD663203437F575F769CACE"
02742     "66809FD25933204387F8C51E0A6B2C4C6113DD45D31AD7567C9B924D5B10F124F6C1CAF345"
02743     "8B3B84E9E42DBA9AC879E2D711ED8681207374CCD299FDB7D7004678D56FFFE0CD34B4858C"
02744     "C0CF4F611D8EB5B47768D69AA69BB1875E9C397EE9453D37D4983359393A02941E772F9127"
02745     "08085F2FC7211DB358EA5D4E00263E14A1CA3430A6E6FBCBA94CBC8281C356B4114B0E0B67"
02746     "B3265D3E1B9A64B1D4BA5501CC0F6E19F56A7B3B1BDEFD8018340ADF7640C58FB26ACE46B7"
02747     "88FCA44BF228A737B02F160B0462846DBED0792B27FC6B55757BE01FEAD3655FA9E992485E"
02748     "2B29321816B04C2BF079BA5CD01CD7DC0CC8CFA471FAEC74C95041B7C62DC9BDFD11FBD4D3"
02749     "B7173A3A92044C793EDB6AF4DCF760D38D3D412B3B292C74E08EF4B4515B99A0D4A167D910"
02750     "3C3E776E213AC2C1614B432A8A7D243B0EB5E7AE82118DE996BB55329F49ED4A7EE738350B"
02751     "565F50CCC313DF274EA203757B372D542D451DB960DD8D7F0C5E532AFBA105C7963A394267"
02752     "32E2E30C68317C4CBB3C10802C5104923E9EAEC89C6F421B8EA3D5039D5175F1EA85497140"
02753     "7C7B260BF57FA912DF3829BA91BEFC9A7A88E5A289366A88E7E6C5C4D4781D583CC15C827A"
02754     "4FC753F66C81B5095D51F3069AF2E3BCBF01362F118690C35D3E6B5C0F41F4A4964226E464"
02755     "6DBD948FE2D1AE837C0C0062F77A40DD02371EA4B69BFD02803AC3AB972CF450F59FFDE724"
02756     "F79DDB7D534D261A44DC4BED2A47E78289BEE615218D1F861896AE5B07729568508B0E35F8"
02757     "2BAD9387B52E2243E456C2DAC3F82176E4859A8288FB8E2B86FFF4BEEF93D9F677CCACA1E7"
02758     "60D045F28C819A44F5F678E61585B74D49B1F12A9B28D4056CEFB98845385FCA12A6AB0C86"
02759     "DAB01C7E8BF5D77618F797288706F18DD71B33F909CDB15A63047D0EC462BCC02AA7E6D4D4"
02760     "D5C3D0C5FA2C03502EC51D781E8CDBB63A2578CDD9116701F7840D6A268DB082223F3D4237"
02761     "6C5D796557BDC568791BB3A5BB6A9F501837CAF0ABEFBCBB500868EBE0FA0184ABFB927A41"
02762     "4A87BC3A1903DDFBAC369B77B5C9FA203C7F6CBAA7BA826867CBD37AFD20E32DBCDCBBE7C1"
02763     "2E3974BA0133B9ACE768AD4652C3FFB01D7E2F4EA026D543B2292419B10DB09C47C110005D"
02764     "667EED5D066D4634175B58F4BEA53CC39F2D1A9623B7C9AF2ED29A37891F2BE24418746A23"
02765     "3743BD340256801EE39E7C05932AA41B56F7DF4E479A365BA224E94D72A395741D3D813520"
02766     "8C96AF089FE2FAB29B9D7DDB02D149275727E12A2358F4CAF9DEF137397AAB54DF78F5CEB1"
02767     "7CA323B696C3267A4D2A944C65B2E8F542A2F0AC3DA7A9D6F2AF073FDED7424942B700AABA"
02768     "291FB8672DF10AFDD154218C0B0088E84486EDA5F21615C06770A2607E93A2ABA60B864505"
02769     "434ACC6B6BB15BD00C3881E59F0B481653BE10929E7EAF4A90BC181DF76434C08F3ED37D86"
02770     "2038232AE6A5C56F36F98574029AF4B4A2806935510 82BEE4CB6DD4B55E00AC4954F2A0112"
02771     "9861DD2A70FC30FCD9073FD2F45B31CEF9724E673032D0C87B2A9FE7706040293B6DFC3A31"
02772     "97AA58A53997BE5A2A763B027D5E14152774E3FFB8",  // PRN 27
02773     "6D562B96C4DB6F8783C945E358693B3D99B7D8E77AD50B7FE9F5176AF257ACF63E037ECE9F"
02774     "C071CBA13080A08E1116AE09151BED458CD8BC9DFFE5744E0CD2365293B365A01D30E894F8"
02775     "616AFDB6115663FA15522C8CFED9E6644DAA1AE776E0BB2912CF584AD1DB53E90043B1F93D"
02776     "67D51B1B10179F0FED6EEB52A127BE857341CB5A223864C02E4AE3F3902C90BACAFBDE2A42"
02777     "D73DBE88577F871D9C2EB9885C7CEF0443EE4A1C449951B24764CCBE716487F2EBE474ED6D"
02778     "AA65E393F1AE17E52B75EAFC245A161AA30CC8250631DEDD2B233D202CDFD1C3DB64FEEB2F"
02779     "515F2568DD7C11DDCCFCDD77B9D1182A29AD5813791DD9987BA735082F8A669F38DDD84EE7"
02780     "C09334AC732849119D769DB8CD79375D1C8EB4E051777A057A4738FC4E06ABC70F8FBA6B52"
02781     "948F91FAC2F2E248F0C417E2E727C60872F905DB2CCDCF15B2363696424BA5B62E98D5E4B5"
02782     "1ADAF38DC97144CA32BB6DF129EAB07313059C725051883475E30B25C8CEDC63BB517319AC"
02783     "DC1E0ED86D5BC5790F11CD2F4B2CD882345706A4CA5863C0395C3159AC3D7FF8F7B5F7B599"
02784     "07CCFAA53EDE296087B04FFDDB0B221B3DCD1A6C64A591729A039C230CA52C94ED2C311879"
02785     "4876F7478E7968F962004F01E8FD0546207A64C09A2292376F77D7789CE2CF04995C7EFDEC"
02786     "91B5F13EFF915E9158F6202D6A7B0B97BB1FB121FF99AB12F56732E9049A2C33597A308A47"
02787     "FBEEE3F530E357BA6B7C98D58114D89C0E9599C6B17BF409B3B63FCF67DE29FDEE152711B6"
02788     "3C02F919FEEE1A31636E36ABB4F9BA607020529814E06AB2D8E4FDE4D6C1A0D709ACDD7E44"
02789     "2B1EB65643FE7D9E8D89906915AA868E9DB2E070CE6511EDA0506A19A731C0DFE149788BCC"
02790     "155B829D52E2921430566F9DB82DB062023BA5FF61FD26F8209421EE8C2C22FAC9F58059D2"
02791     "23B4659A23BE538BEE894015F8E137849280668B971AD628CC57ECB49701EA1F0CE2D07B2D"
02792     "7D8406FB1D2E010A51C02F6F8F04F38703326711B85ABF70D46D947A42855B7F13B79D39D2"
02793     "22A3578D6336C9B8A69A12DD38184EBF9DCF787C0D7611AB4176294C239B60351B62309E5D"
02794     "BC92BE0C0EE1422D9074905131757F70343CA00ACF2CF9DE239D8C794AC868565FF9900A1B"
02795     "9D187580884D509642AB4F6331A89FCBDB367067712B7B4A7A32E266D7FE571E4474FEB5DE"
02796     "28CC0D480D4AD11D0CF6B599A56925D68C1B076CC57612290221AAA0F91C8F7D21EAB35431"
02797     "0250A224BC0455C760F62B2981ED68F7E10619AEF954B79D0D0DCF039DF951BA2245DEC02E"
02798     "71968009ED5364774644E995DCB33A41D53B3C9C1BE952449CA8E671E15EA888789CCDFCDC"
02799     "C91E0ED093CBFF538909886E27972C9D262F5E1A481B8F483BFB311D1595F51FE1AC8CB43E"
02800     "BC92C0D17868986B6F2BA8127BF95E159776570E273393FEFE203FB079780E675BD62F475C"
02801     "C0D1DF76CCA50B5145565C3615498DEF9B7BD37887BC6988A9853B4E639C1BFB61E32954D1"
```

```
02802     "166156F944C722444C29AF577C12F567E568BB1BB44A9D0783F126D6A079345C3842AD6240"
02803     "F1740585102F558C5DB6C22A057EC2D902CBFED926BA5F9C298E6F9118B9D2B446A47E4DB2"
02804     "F4F1CC30E75B6A1B67FDA687E672E7136AED3E02F8CC346F520E5929526B6E9126450E708D"
02805     "30168D1A60B41D19949AA4BA5CB7A691DB90A7B5040B01518FDDB6035454280BEF79E019FB"
02806     "96D899FD97BA47D0A7EBB39A11A84036659598294B0B9A905CE2E0CBA0E3AFAC85FEF2CD34"
02807     "46057ED5DF2FC389D411CC3128A8DD23188F830D90",  // PRN 28
02808     "3A9010CF2726ED13F833686B6C7796E87CEEAF4A99B40BE702CF35774B6BBE4EDF528C39A8"
02809     "FCB1B04AB9D25FE6666535391A45D5DA45D8073F705184427C3C5C4225F66F041151A22423"
02810     "A1F5516C744FDA91305B2B70BC11569416ED456BCBE6C1A47A5805B464EF6A645682D88D75"
02811     "E7FCB458BA2F9C44E14589D2805A904AF02FD3693DE69D0CAC00E058B26277018C3339EDA8"
02812     "76CF820AD6C99149BF76A8C47ABF795EEF76D55862F4A88B26A73385589048B2E22C542461"
02813     "AA9CBB41DA58C7A5A32047B9F85426119402E531B023E3C8C1EFC7AF70D3D553B50408AB56"
02814     "8E4D2EF96E5FAD8BD3EDF830FF2B12829E97C5C57927E69A438737B1920553F2B921E62A3E"
02815     "25331BD3BBBF99C9FE1912E0A84B15EECAD16B03D059876B7B02908C7A7DDCCC51E01213B4"
02816     "829D21C9FE65AE129F3B58B51798C604C9A2D2BDF7376FBB50732D9694E199DC532623309C"
02817     "A43D837EBCE0EE9F3943BF57F9ECAB4D939B2AE599FF5CF5370C9B56B7B9800A651FA64F85"
02818     "4962A6D84E1270EFE2A481D399DD3F35C9C43605FB03315259989C4F2EB41B83C62DC2F7EA"
02819     "15B315126D227A0CED8B626022F9F665B49B8C32A10BBCEB0AC7FA80335A255076BEEA7E5B"
02820     "EA0472531B22E07A6579F9726600E661B389A221D2A6D2AE9E0059D24874BA5AEF82B1B537"
02821     "34F0835FAC5732FF0AED0F092E06EA84B3B5191BE3D226A41A6BC39AE3064A789A8DD9CED6"
02822     "8FC3B902249622576D4086D63B924CB0C309F75F1FE9A62071FD2629E927A3ECCF3B575961"
02823     "221C5C1E76D2D6F1D330028F262E671905340C7621150074C49D9D3D64563E4E6C8B3221EB"
02824     "DFAD5A369E25DCB2C95A1E19D05B4ECB8E605BBF49623F6C62383A4ABF25FF823CB5F924AC"
02825     "F6D6173BEB850C46E0CCF6AFA873096046B7CE53271A33D18BBEBD6452D1874E312B9DD203"
02826     "5E511FA43BCBDF3E3A9EA8C602EFF6051AC0062232BF8A2C1AAAE76F0C12ACF4A290AC376A"
02827     "0DFA55BCECCDDFC1702D38F74BCC3FDB3160C3F52518EF2B1E14CE9CF22AE5E4CF2DFD5310"
02828     "57EC85896708C8BB9D810B7D258AE975D678F19139A760F5609F57E2558E1AC0B6AACA4636"
02829     "017083D25190CECF73D29E2F7B44393D274074A68570B720EE6CEA533AD788F9A7F127BDA0"
02830     "9D43F96AB9863B5365C9361A9EADA698F5A616D5EA0EB9F83B0647698DAFFE659979F5A5F0"
02831     "86FC8B59D53C737E2D38EBB4F74C5CC6429B526FEFC9D1426969AB14DBD3F54BEBF25A5A88"
02832     "46874562CB282B2BECBECEDAF015F41037486297B30C905572211F80727CEF292CB2DB1181"
02833     "57E666B190809A10B177BC63158505BAB5C915D37626EF7CB011041B08B03FA92A40EC3E39"
02834     "BF9B4D7279A5D1E4CF38377BE6C00EE12CFB4549C036E75240D3DF9FF92B50954F461BC25B"
02835     "D155AF84316A1752B9F1834AC4DA1B1B5A4C84992141CCFB0CC6AC5672753F4877403FC3F4"
02836     "A52848786C976B18CE968509A4014F3C69A77F9CB4915A1E389C1EA74A05DC08B6A6BB1BE7"
02837     "55365F1E96E06FCFDBC7210099601912F55F639C9C96F28460CFC392C855CB2D3986989FF8"
02838     "5A0DC7889AA2F6AEBD3E1AA196F97A3C3A0FBB33ACD3BBEDFEE06606DDAC5DA26290149E63"
02839     "61E8BEEE7F2400588F3FF853EC85B5F5742DE3547F362B0A094D6F935DB0A2B5D1480DB2CE"
02840     "FB9D665887F6030358AB9E5B26DE8955B188C39D6B146063FA59D1BA260796A2DE8CDDE856"
02841     "0915F07087E84589340103CCF7E1B75AD5BFD209A99C429A52E8D4FE96EEB9A32A4CD24A79"
02842     "40B446003E1F8296EAC6653659CCC3C85E01F7A154",  // PRN 29
02843     "59CD72F81F8FA04AACD73E9C815A312E8C02F1C1769FDA17C16350D9EBD4F8D10F248035EA"
02844     "FFB4FD67933279518E3BE9E1807288A9D63A6504F3A31E7DF322AA1B2A9ED3C23717C0C5B1"
02845     "34D2758666CD04C2FA8C1EF5C1E58870C5EE1378619851845BA80DADBC37A764BA17E6DF28"
02846     "5BB3DDFE31A56D4BAAB0FA622B8D3D24B7F91034D2F773FE578C874FAF5204F3DDB90B8533"
02847     "906AE05834BE67263447FB4485BA069EAACB68806028CFDAA8FC6DAC6383837F9A059EBD15"
02848     "29A92F98F44D55AC06083D6B751FF8F9FC5A6444C1A7D346025AD4B67EA12439242FC02D22"
02849     "8CE537AC9044F6D193E0592684FAA4D259DFB00F5F98DCA9FAE4C0D5F0427C6936E501148C"
02850     "56EAD1817A81DA30989D0D9C85BF8D9972FFE67AE3DEB2A8959D63B3C3E7737A1851EDC8B3"
02851     "13D75AA7838C6F7750280FFA6321CC9228ACC596431D72AC0302582CA56C6172599F30D0CA"
02852     "04E71F3FD7A03D467E37A8320CC0FE3AC72099CEF7C4265F7F18E2980168DACF9C058E39EC"
02853     "9FFE1EEBDCEA5F78D63BB18BF0544651794013460D4CF8927580B9E0370B2D27E4C6BD4687C"
02854     "F0D4348B04BECE8424A42B2CECC668DF9809F580BBB3F3D59BFD96DC8C2E57EF5D2989D6DA"
02855     "54C04B15F85EFF639253DA3301FB6E6F92264FE48968C2C2CA6D40C54C327E0673BF769CF1"
02856     "16DFD70CE374BC447AA1B121827E5BA510E387974B19B050D9909185A514BCD8CD374EFE20"
02857     "89290D3DF46D088D74F556A8E5F36B08AB17E54143B6FC374BEDD6FF88AE74E71C72297A31"
02858     "E13D1F668DDB455DF00DC91DBBDB2E96ACDB811ADF7E4C1C460D8FF1FB03BCF95EC489446A"
02859     "8DFC5D2220E58CC298FBB7A9B8EB33B61F6461D77CEBF9C36BBA32B34998B7B9B7950E7E98"
02860     "857371003F40279FC2623F716B808912651E55A04F9418989047331C454D2AB04FAA192938"
02861     "209261029FB3C070FA056D2AD52BF9C7A0535FE2868888BA55CB9389E358CEEC68D7A0E550"
02862     "5FF0DE4263C6E47406EEE83E8C797BF58EF49F2F8BB4EB3656A2636A5EC604A4990001190A"
02863     "8A7965A36BBB41019EE19C8EB45C10173B513F617B9688D7102ED82E60DDCD080233A7B3B4"
02864     "3B641A3282C4AF6D3F01BE30DD78AA184B9D565F7B1F927EDE214E363A18547263412EBE35"
02865     "E939942C4B5EFE81E439B325902EDD4D2AC06340296D00DD5F0908F5C8AD88049D88282D0F"
02866     "2DEC28D133E1D4736869C352A7CD87D8A56687CC25D5517A62F0CA5A023E8709F181A0D96F"
02867     "1154B3D7517EC5B076B7C59DE90A223B96AF93F94B11EA0EB08A280379BE028DC494C296BC"
02868     "46659CF9C26A3F8BBB47E063AA4EBCBC9873CCA2866865CF45076E6E642B8E3BF9F4A3F9F0"
02869     "57DCE71D4D5F3B621D8791CA5333398ADB31486D22F608E9A8745B9548EADDA654F1D4BC04"
02870     "2DFF9B3C21B15AEBC37050864E4275D5C36F301D533552D119039242BCA25DFABB524C9718"
02871     "21B2C0EEA602A4C02C6487DD113DA4DC5ADD7BBBC325CFC896C942682945A1BE921FF71E6D"
02872     "76FA1E3345589B53C549BD5056FA01BD688D0A9011497C0DFDEC20210BC7F8B713C46A100A"
02873     "396A9ABD9A2BB94E5455793BFA28E39E130645973C30FDCDE52F6C272576291F0130474DC2"
02874     "E3C30EAD54818E9F531A538028DD40FE01D993DFCFF82679DC3C66850528A173FBB534843A"
02875     "B540E21C7731C65044D697B42694A1654B3572FE321F61E98EA22171580F5DB90C997D0C9C"
02876     "7CF44D174E8D24F500832C158278CF62BBFC6DFC46AEE5DD3F8906A5770C3BB6CC3B731243"
02877     "974A76C0BCACD9EFE37E52CD9751894A68AE3596E0",  // PRN 30
02878     "C0211AF7A94B37632C667AE5097F8B5992917210A515C3E48276A363A6F5696AFEF5B39196"
02879     "89CC8F21E077310FDFC5A58D6BD428B8FFC678299FD053E7C016EF3A154A41DE5FD3C3BC50"
02880     "70ED6EC8F16170C880F82EB0B7EDED047436E8FCD4E2290F450963133226D17D7BB99C7109"
02881     "7978CEFE496032EC8C6E73B99B05A33C4F5373BB73CA93F7128A9818197E6FD2BD88451A4D"
02882     "8905214157C4259228DEB5F8B216FFB206E4EE1025D3E16987FB6064147D3E99CEED73112F"
02883     "62D513C8B6F19B582D0F6794B307E8AF97354573C758AA95AB0A926386E7D288D74F9C6B0E"
02884     "2C71EAEBB36422C2120511F5F3D2B308674AE355B1FA1E72DA55098AAE3C70CB1B85C902C1"
02885     "03056E00EBB70D37F08FA4D90E860C74E841F4E99957B3097ABE602E65F07B8787DA172E8E"
02886     "9ABF9ACDCCDFF6FEE2F902082BDD49547EFD95B54C3B8F6C6D243D07F16819E03189F0A3F3"
02887     "E0D05B3AF3EC92CA08846FC909E809CE9C30F1F3AC6FECED9043245A00590CBBD83EFC41C9"
02888     "EE3CBD0E328426F821ECA32E0E84A8B64F200DFAA5F01B262F4E816A90AA33A2984AB8E269"
```

```
02889     "B8CCF61E65094FBB7D10E11F9E1022CD97F17011DA53455B24E4ECD0793486C1E6D4E3BEA9"
02890     "4E6DC9C710CFFF30CCD48E43076D763957260D1ED4F7C41EFA0B7F78BFAB7A9274E932013E"
02891     "605246C36CA1FFA003A4B24DAED61D15327BCA1067ACE961E718DD5E57667F5F33411D17BE"
02892     "093A9D7C6675097FC551014545727BA3F19C5D5D23A0922B11C4BADBF285A17EE96C916238"
02893     "C1E1DE8A0D28FB79E72EAF5AC37710B4E6550CEECA2F22697DA18F858507CCB95BB7F25A00"
02894     "56FFA55EE9564782C6042686195BF56901D4ABB3398C557CD47D30EEB344B532E775668876"
02895     "93CFAE50CE8C0BBAA204465FF33456ECFBB1B4DE4F8D6E2F3E8A84E90410EE3A79ABB47D40"
02896     "F4691CEF37E0BDBE1BE386098181EE36051EFFEEB0F5898738E400D44ADA4B3348315E0BE8"
02897     "8C4C43E8CF95CE8D56EEAEADE754A09F170D3ACC44DB101037E001DCFBE2C52DC32D040BC2"
02898     "7C1A55C9EF5D65FE418F3B761284C461606ACC84D36B9A7A3C9353723BEEEC80CF75D84DA7"
02899     "B2533F844E81EB8835289487CC7EA4EB277A4BA4BD50B59953E3DF3EBB2BB69D7E148792DF"
02900     "55D429B2DAE9B40B00FB14A948F2797B7488EBBBEB4B11880F7D3062D3C0561D7F6FE715C3"
02901     "BA6BC703BAC33C29DD87EBB84C178BA09961F2C10D8CCA42BFC3977EA66D039FC24CBD2A7B"
02902     "3316EF37C0D0F98D3CB8D5E4708B4287FEE6B656F1E300260D5643E9AC323A7F0527A7B981"
02903     "74AFD3277D35D6EC63865BD4E816351B8492474116E2426A3298606039280AEB595293D18E"
02904     "EC671E1E139EFC7936CAE3056F9698BFE95568D050F281F17F2CD65B62B98BF36DE469474F"
02905     "440E27073613A121370C606D5729DDFA12C8C400D0364251D850D90804ABF000ACE57AB195"
02906     "F6EE24163E8C443C8AEB641AACABE70675ECF22A5B9DAB6DDF0078EC89F2354C27D2964110"
02907     "41432AF390DC71CC6141B52C54FC778B5BCD31B55EDDEB62EE364983E97BBB65EE98863AFF"
02908     "D2310B482AFCC28A2FE9C7D8E3ED24D20FFAD1F5B0AADB344C77F89FB88BB2FDCD64A77203"
02909     "BC331FC23BF78E284ACE3016FF570E3AF32ACDD5BD5E7B59115950CE804C231AB1AB802768"
02910     "42FAF80A576C9A3148F527596252E2F9F76FD6B4C03EFCE924A4740EDED98B290FF1389268"
02911     "11607CC553FFAD5ACE0FC24D7FA1D7F3701A8E0D7F614941C9FB9215389CC5EF6D3CAC193E"
02912     "B15798CB1FF2950E2E0EA96A6B9E672CC343A7FA9C",  // PRN 31
02913     "28EB964EC60708D60D921BCA637AF0F0E19E127E0E563EEB30DB02F8FBF2848CEC8E351077"
02914     "A8294AD441049000D2A3BD101B691C2F1DF3741B8BE7DC57592C2D09C9F368010B4B3922A8"
02915     "A622887C37E959C9AC31DBE89C9EC9FFC26EFDD9C49362E3400E660928BDD7F67C6E45614A"
02916     "DFFB96622390AACC03F1186D18BD11333AF0348125A92EBB40370A32F5E383ED858F6720D4"
02917     "3633A9A3E3559A0ACF5FCC326DF117AA495759FC12398B2E060F8C6DA823F3E58E1158B81A"
02918     "FEDE50B89F956D4F63619332928B4CDF5A16E0536FE7426518B1C50FBBEB4EF3830D77924B"
02919     "E1DB002BF2B0A9A8CD6FDF3BE7D7D1D7A3EB4E8CEFF8BEB9E2CEA9BCA74349D67413E7D05D"
02920     "EC014D1E6990062F0C4CF1E4720D45E23719D748D0C9BE3C3CA444E357F181E44EE841E35C"
02921     "B5207C3F993CF224A4A90EF5E3610254EEC4C2F2EAEFE041E1791518F93BE9080634AD5594"
02922     "B98C6D0161BAD23EF1403FEE6FF846D344BF0B42A65762804A33F00F12D769B3C58F123E14"
02923     "B4001CC631049F6E35013DF92AA50F89E71E78879A9B428341A738CA001E4233652802FC16"
02924     "AE2377C9C0B5650805205D31445EA0AC0F82A109E5F7C0758EBD5C737CF48D3D5D44816504"
02925     "D972CD2FBBF0AE566B8247F81E404D7DAB5A939AAE51375154A3125345C061CD479F8D7D9E"
02926     "B21C6ACD971F75D8430FAE557DE7A669C59E35DAC4E7686C53C34AF64A5AB7F6B5ADF48585"
02927     "417DBA3584D4552C241DEC336E5EBA84C4D295148B29FE824D3934F4D2A8EA2D27D2925598"
02928     "8396A5C350421E2C2BD89B5D18155B20260DAED20E3DFF52F2E84E6D6CC2C040FA3D0B8726"
02929     "C63778A601CF95BCFD381FA520F95146292B8D854C5F057440833E78057D13FFFEEF4418E9"
02930     "7986A064AC8DB74A001BCBF8B8E26CE9C68BACFCF4895C956666F92B5EADD27FF6FB032207"
02931     "FCD2100FE0DDEAAE8A860F89BDE341E59671BB0368B1C68B44B131139A44BB37EAB251A033"
02932     "9A79D358D73281325A594E2156E981C519419D925F0D6C8661F7C294D91F527454AE82EFF9"
02933     "8344A6E39B2BB3E36431255E33DAA31B75E961FB306792510924C3E9B89150691BDDFB275B"
02934     "9128677DFEAFC89E6562FA245DEECEE2E788EFCD06C4BE744CA97693CC1AD7F0EE4C633D70"
02935     "6AD278A46A8DBF2DAA0CCB69806912F8C081BF192ED7D17EC222392B6E84D6CD88FE66DB10"
02936     "B1EFB9CC7F3E4F3719338B8D6BBE476440009743829F288A194EB3F6F21FE7D7FF55F8FBFD1"
02937     "312AA603CBB0C011CEA377E6E2AD1541C5BF55AB73777BDC20DE77F701E7DF26728A44D881"
02938     "EC44E905608CEAFEDB670A261F2109DBFF79CE13E98A3519A9A86F3E01B7C3EBE2F01C657E"
02939     "E017F0740328B97392C6824FDF4D5820458F0DC9E999A4F878D09493D249D0B43268D44135"
02940     "C966B05AC8A69518BA5AA10423615DD3BD8B6E38D37D4E7ABAD0F8752A64736EB4D6656F6C"
02941     "3A6C6E1595AE0ADB31ADACFD6B4146F4496E739C4DC98835CE4C5E2F1FA899D4B896B02CE9"
02942     "7822E90CABF5850321487D0C9DF07772031F1A438E9D63374373958DD9708C6BC38C3028C2"
02943     "C7DBC7B2A5A0CE8EBF87A72EA394FBC0046B62D60CFCC4256930BE65B22305C750B2A7483C"
02944     "BE0B3FC0824A011D274FCC289B8EEB9ADC5505426D8618369E0135FCA2FFF5456E48010995"
02945     "2CFBBDA22758E14643AB48F11AB4F4EB475CEA238C6493166BF384B1DED53072D897E368C8"
02946     "BF8239DE53290766A4727CBBAB88691400E9FB9CD53442B91B5D421323A24571DBAB65B7EA"
02947     "7015775B6C8DE25247D0E7BF3DB7F2506B4384B328",  // PRN 32
02948     "D7554B01374AB97EE1246C2DA82B0AD5BBC76968B6E058A821418E0707D596E2A7E3AD9AFE"
02949     "623D18812F269C1D347A732BECE58260D2E8F1399AE5BB928716865B902611DA70761A4C3D"
02950     "B3EB0D332E58E081C08D4AF96B1A85DD68CE15A8EAA5B95F73CBA9975924F8F9A3CB57249A"
02951     "6A904A33F6471ED4F389168702EDA2054B4A523B6CE189B26C50900913CBC7693C4E447CAC"
02952     "4CA7B783106698CBB78F5E793284F27E182CE12BC792935FE38CEAA7F659967111AC2607C8"
02953     "316FD3CFF111DC5C2D58502555AA589010F9DE4F735D456CB9F0896FEB163984710BCA11C5"
02954     "D78D9ACE6CC9E2EAA70B177A9D33D8C59245FF9773AC181548BC82C751D59A350C65C895AA"
02955     "79E9B57FBFA2D17F53EFBE0C490B43C3369B71DE45B9AD67C88B2A8F1AEA2F5D871970FE9D"
02956     "4C6D63F81647DEA310F76DE6C050729DA4575087E3AE8F6804128FDA47C1867A6FB5065C41"
02957     "86C47D1199BEDED29D26BF9825676FFD9A98BF91E3921A07EFC949D7473CBF04E6647390E3"
02958     "3CF765BBF73714E0AD38C4D1EFBC3BC4144CC9AEEBA0B9B4843F910A7F9CFDD347F6DC1017"
02959     "C010DF0EED1A71958CC5BB8D50BBEC164E186B6572FD09715C29611A3A5162496968C1DFBC"
02960     "1B6A8A4E617922D152FAF2B08BA40E47AA0E9F30515C9AFC74A5C2D87828BAE5243BB7988D"
02961     "ED81731126E4C7E9609863929F943930FC72D0F3A4784990E08145485CA0FBCD69BAB2391C"
02962     "82E75BC2CB2DCAF0E94A3717B6A157070C77A8A3B5DD1E30AAB09F5368F46F66575599C042"
02963     "96982E6AE8F8C334ED24CC58449CAB0B7660E0F2BB9D96BE515245DF9654EAFB5060D21291"
02964     "DB8A2D820BF372E795957D59ADED6A72C265D19C852312BFDACCA65431B9AF4DA6AD56B97E"
02965     "9FDA9AE4650C9165037F353D9BE92322B4383C30404706D102F15D8D39B21D22E0B23FFB0A"
02966     "A0F58F477DEBC4F130BF501C765295A54913A7A0190FC907BF2420B329B97F30FA65A7429E"
02967     "7D71E572E1BE0AB0793B606F480A1526085637BBF89CABDBD9D19D808F230AE48505CDA9AD"
02968     "E68FDC826682BC406D4A7575EE913B60769C3BE4D81A073C05FE2BD6D7A1C3B52FD2F22875"
02969     "BE856CE0F6D7A3AEA1897FCDF951DF4A86184E717B4077A25D37935A82E073373ABBA58B0F"
02970     "5F744CE52284D96200CD90F0A7B4A03869A93C9371B7F946EAF2EDF35ADCE873407C7BEF0A"
02971     "750CEE48BE3A33CC47979648FBC5EA7E79DBC725C3901D4D05BC5B804007BD46E48B424665"
02972     "763F028E3777FC1A00F588D9A26F68DC7ADEEAF8214B747B409395016A05D8C72FC3E838DB"
02973     "F7C5A222D03699897E5606433CA760C6D67C8BECAE0FA8C312C9875AE26D61FCBD3A320829"
02974     "B615FED0587E4225E4B62264C9B37D2D2B45FDDCEFC060257960DF6A1AA2AC0544FC333914"
02975     "D192A1629061730213B1455E6FD7A712A08122C8BAD77EAB9CB9687AF97A3AFC02B8F41393"
```

```
02976     "0DFEC10DB37C385B5643E91F3345C73EF353453F34A5EADC8EB41CB098E9243E57258AE608"
02977     "C9FE7BBEAC524B21A0734D8EE51EB30DD6974B87FDF3B0B1E1DB01FEF13061E4B072DAF927"
02978     "C99BDBB7691069E2493E86239A69BA33AF82998CC1D7B368A4343BC63EACEED280415FFF40"
02979     "B0B61221D7D1C098C6391E5C888228045494F86644F34AFB1B191BFCFD1A29C678956DB16D"
02980     "0BBEE458AC6DDBF00E738198C093170FB37F3F141619E65502AC57E9042CABC72372E5FAB9"
02981     "DFF5E42A00EB023E44B4E536706FD63BFEBCE1C0DD59E465A61DABF566EE7216973ABFF5D3"
02982     "4A5A62518E12E8352C61BED9348E5826290EC0F84C",  // PRN 33
02983     "4251265EB325D261D721A84F62224701C9F564298EFE0C2516330F7AA43CC3D7090581B74C"
02984     "69218D74C8C6563FE276F24F39BB7998CBFB9C49EC649106DFA51E45498DF0F347DE8A5623"
02985     "41DB34E94F286F583832E49AB38C09DBCC3B2C307188A980574461CB3C41EDFB933938F89C"
02986     "88F61886B60097F272E511B49CAED645190E98C3AB8AB5CDDC021F76567AC1EAE9F4F8D782"
02987     "B91BDBA3037BFF8FF43879FB6598D84046CA06B3C8EFE275BC61E449853F979AD6BFDF27DF"
02988     "453E2551F09FC1916F4E9884012BD4B946E2A65C86C609CF8EDDEDCECBE9B1DD6715C1792E"
02989     "261A3A013D64F00118DDB1FDAC9900E8BCCCEF9246446D5248D5E3423D859E1AA9BC38F869"
02990     "7F8A4EA08632983F268BF7D0BE639ADFCE6625E00229FD510A41AEE418C3AB85F5D2256D8E"
02991     "A4E9AE47316BE6DCF792537F28514B99BD76D098C8B4CA16DE1A9ED0F2693DFBA9B460BD84"
02992     "2452BF563B03874DCCCD3E18B76AA3E352978473A2E02C6398438A35639457DBA7316577B0"
02993     "863762FC177010E0FA0F7537420C0443D8D7F51AEE4F40A67E249D678E77F17503A0A6DBEC"
02994     "3C60FAD140DA592082C334E3B1D021E34104A9A724FB6EE3B5A22CE5D3BB9C5E896956B4C6"
02995     "F7B1F51A74A56A7232E027F706EBB999708BA8ECBF42F44E41EF63A7627C60E5E75C678B5B"
02996     "3F7846CBBECC9A1B6EF78D7912AFF0EA56B7E25C31784DF9D63398FEF53F6193554C4D8B7A"
02997     "96C170A89961D2654B9A663B39C76401A439CAEC482BF44F20DD12140D47D0580DDA48EA90"
02998     "5D409BAF5BCADBB8394333B97C0FD9501AA4B4BCEB3F041520422530A09A9B24EC04206CE9"
02999     "07A33D0A18CD5C1640CFEF970623B1AB2B7EA5342B6F38C717416C17545D27C21A8F37AC2B"
03000     "98991395D3D170FB2D03C6B152915DDF9CAA12B5EB19E45FAC4CFD4ABA726A723237A64887"
03001     "73C05D0D38D38D920C6E8F2D3476B06B40094463CDAF380026BC36B8EF3DED6A01CA89E015"
03002     "A0B75BDFB97D859C620D329C9B26B1DBE3A3C92D8510B060ADF4151AFEED125ABC0D6FD3FA"
03003     "9F7F3C67B6BAF14A672977F2052A84F0437027956AAED010F63471C5848781BEF8F2BAB9F8"
03004     "CB0DB00253E2C6FD30D2A0794FCEC489376261CDD407B4ACECA129D87FFA0C76E28B948303"
03005     "4A40F84EA21E06C37BBEF2987AC16E77E021D29CBBDF077A912B89CEF82FF0EA59F7FDA55E"
03006     "E99F4602B5437CE71B4183BF28BE79327483898C39D09DCAC59C3CE32333BEF4B0720B3094"
03007     "4B9998A01CB3C135D53B4A76FCDAD1C321D39D4466F0BF95710FC975ABCA1181070A4A5741"
03008     "624CE9AB0FEF27A5BB1805BF295861D0208BEBE7DF92EEFA7FB123B6AD42541C1A1057DA25"
03009     "60469A680E3664B1455D6851E6C8DC8C4940281AFF025B2BAD2CBB53D1670641EBDE2AE91E"
03010     "71F7F69AFD0C1A0225351CB1290EE40F81560A52463FB04E1364F9F3E5C454C311240EDA5C"
03011     "6AD78A094A73DDC1463E19A8E12419DE4966A1F12872E566F178E256F257FD1890DAE6F84D"
03012     "E013C2490B57DC1DED2E705C0134E889EECBE5575587EDFC240DE938C6D2D9CE543D5BA3A4"
03013     "A3EADE1A6A5F0CFF996311F72B3C6761414E34342C3D1A48F5D3011D7BC7F936E8E086581A"
03014     "C256DC6A376F9C4ABAF87FFAAD67BAC64401F4D976129EFD57102A84E63DDFEF60CF13EAF8"
03015     "973881FEDEBFF8D4B356B2A03EE9FFBDFA99F77EF7251C11EB1D2514D5E657697EFC47FC42"
03016     "CF10FFA7F09765E2EC39E0A99E4FEEBD410BF4AFE51855341279F7E47DFF0066602618ED5B"
03017     "F70B2F6D002805F41E991C68D64ADEAB7374F25B60",  // PRN 34
03018     "E0DAFB980D888AA2580DB37930D1942D6D088FDC706871A90D887B5F2CA6B21911BBB8BFEE"
03019     "1376F484282386O7AF7CF82C3B3C9223746BA9A5F01523B20889ED8F139EA3D505D55D766B"
03020     "D056416193CACA6D5D6175C16C38BF44D7F89D4A5E3406C7944AD05D86C310E1EC78E0C8AD"
03021     "80FA0DB202AB1768AD446AC7FEC60B30D741114B45857FA5223E64AF32476319F16DE15C04"
03022     "F34F4730AB31F4F6BB13252D706F0B3BD5860BEB864618353CE62469E25CF4AED50D8BDE96"
03023     "33CF18F76AB1EA5552614D4AC3848A0CFFFDFD51F17A4E744533DC52979D7012AE53E9B08C"
03024     "4980AF8876ED2FD8A6F174B3AF651517BDDA1965264F1CCF0EEC8E3A9969D1359C5C9FD20C"
03025     "AD8698F3E28ACE77001F563BF456B5A46FC747E5D7EFED93314F7AA87CFA60405F968A4F97"
03026     "74E30E2266938BB19E9265FB71BE0D40FE6BF14FE02E36FE10FF110D63E02145C78B73217E"
03027     "ED9DD87F70077E4229572E6547DD71828C773E46462E30B497F9F5F99D723D657E17A288DC"
03028     "98724B11AE87B7E56D33386A7FDEE0698353CD46409FE8D2236958B166B1C697BB8A42946D"
03029     "66E390A045775A6FE7DEE4BE260490B69C56476B894383091C827D02E960D9359E8D3E151F"
03030     "50EB87FABAEE8ACC6E325B36674D9C7CE249B879B3FDC6AF95E92FAE53047BC211757FD125"
03031     "64EDDA4DDB6AE8C94DD3305C21970E90BE05ABBAF912ACC52C77714F87C2B059A9E21005D8"
03032     "903A604D82EC22192C46CEB63C588A219CBF755B5D44CC0E918CA6705156CB497FAABFF637"
03033     "45C826FE16F65582D4820141DEBE3B949C29EEC6D99C2A4EF34DC878DD82D0FE6D5ACA302A"
03034     "DFB897DAE78EB0BD28A938D6FD5A3535CADEDD7C521E45AF4BA92FD40DDB009E35C21254A6"
03035     "DB959698303D56B59D8F319F18E688F9A20DAC8058E0496E0E46BD01880F3C78EEFE98A511"
03036     "768919116A288CD3CD28098BF5D4719EB64D548A6D791F1A9A9A3E9B08E7184AEFE60A3D7D"
03037     "7179CE47562E9133068789BBE0297FDA2B2D6F4248B288CD899BF3231EB360ED45F769C016"
03038     "7284319D2C2B11375C2B18E5B67594CF499A6AF2687E0C1AAF09F4161F3AA8369BDBA68AB0"
03039     "22459A2DA9DEEA13E509AF3A68D011998D9C30BEBEBD04BC9983F36FA549BFF3E3D323722B"
03040     "2136AFCE0A2CC8E593B370C7B48635926B9575A77AD7C0DA18E94FF1C8FABFB57B25981288"
03041     "E1771465360E2D619272CF3F56268B7EE130FD07DE29215FBD5D0A72B73081869F60CB149A"
03042     "1E4871150778A2229F3C886200E0DC2C76AE049231FD3171E49FF44BDADE185883198D2162"
03043     "9CE0AB252C7451398E0D411CB01A66BC5B141CB150BA9CFA9F9F46339625E1D5127998302A"
03044     "FD19D5B2D85F3F0B2B354C7CFAACA3D4E7C7B7DF44E22EC2C226178DDD480DF7787EB28F02"
03045     "85B1F973DC47B094491DDDC12BF1CAEB99E0E13781FE8A64DCB0A05A30763A055A6ADFE543"
03046     "4D7694A0F6A0366B5D74E73227F3D503E6B989C5BBF9E6D81EDA7C46D890FB2C5CB439C5B4"
03047     "9FE16A9BEB14B12B08C47713331E8932D4AF6B85444E3F848C5AFDC095E984449F0D126EF7"
03048     "11F8B43831755BC0FAE8D9FC79E894CEC183DC24B1E3C81FAF6A546E464AF8C0307263680C"
03049     "EAD9ED6E80217F98CA82458BEE59D2A4FA76BD7D4425E0C77164A50A2933972108661820C7"
03050     "5EB51C8A0ABDDDEB30F31F2E575AB9DB723DD084656691096B72193C52053EBB8B72C6B30C"
03051     "49185166FC39F3D9F3497606D15E8D696CDC256DCBDB6D01DD2C33C0249D6069788DEC0980"
03052     "2630597C13EB1AE152E8E84F6A4BF60BB09E049EEC",  // PRN 35
03053     "EF79F2E35DA357E9C076B00C6ACE7C50F865830698571E999E544F5025A99C36BCB025B796"
03054     "7846306FADCA22B8BED499BC9A80EFA7BDBDB83B15DC6D48AC4E3978D528DCE0A3E11C3E62"
03055     "B619EBA0C853EDB0EFA1DCE9EFB2E024FFC0B59A8AEC329FAAF418DF9BF70A250F2F6409A9"
03056     "FC0153161F6E8713005E53154DDBE15242AD801C216A1E5CE207757BFFBED75EE4C96CA0C3"
03057     "CF448966540EB434C68E9A4CCF3C5907DA216BA664B073D73B0779204EBCAEB55446AC6E2C"
03058     "40B8BDC666D4C9D33B7C644F867BCD8962274630191EADAEE7248398B9F02744AC5D4DEBDA"
03059     "B2D6D9F8D6C6392683FC1A795F70134E790718D56DB6D424725DF8AE7E11F02D684DA78ACB"
03060     "4B9C311806D082C1FADE4B31B8A6DC0E5F16618C47E03E8D78DDCCC6ED869F77CD39926945"
03061     "466369CC2371D3B51F58DFC6471D742C9C82B1B7A2B1B1BFAD15FEBE592AC0F41B19C126D9"
03062     "23BF5944408506ACED28FBED161902EC5FF19CEED25D18F47A76860777A3D86B3E0BA7446C"
```

```
03063     "3F1EBDA88F8A8E182CC0938384842D42F41E0042C53C645DC87C88429E06BA9906A7A3C1F6"
03064     "A76671D69703496324406C7FDB20B9C3CE9B5C273A743DCECC0016BDF85D82112DBD4C49E2A"
03065     "6A5DCA22738750EB8DD6AF15B27A0601970AD12660A51C560E5CD44B55960AA266A88B82E7"
03066     "61B53A8B42E7332EF2FCCE093BCB06FD18D916FA10F4E9891B103DDE5D808963E6F71AE747"
03067     "9A5D0BFCDB8A2B2086308CFB7813EF6B7B5304225C2E9690E47CBFCD534570933541E8DC99"
03068     "227986C0CDE40924D64710A7FE28D248E667E18197B44D83FF1A0691E9049611AA1E04A1DB"
03069     "06595EC32A9B91C46F50096553C0B5716D2EA9C9E3D373C787783B50F28948EF96D21FB323"
03070     "CB69FBA26E8C869C745FF08AC0C71D2B656E42928D09AB4274C307FBB04AE41FE847721D95"
03071     "FDA55510893DF24F56522EE537F609FD54E4DAB4E98E9F3F8E8243C5D056B4CF6F23304085"
03072     "14C6CA22DBC79BE374E2B34B3E9A9B689B9375035F5E32013587C402C14997741654DF70FF"
03073     "5F74C0DC26D82AE7F036731AAE15D596E426EE2E7383EB540A48EF41FE2172F1565AD79202"
03074     "A016EF31255DBA5DE29C8C77B4704EF177C97DD507360FD01B44D612DCB18D7B5650032B1E"
03075     "768851C84EF794F833F9D5D198603224016D3FEA90F3C65DA465418750A69C43050A738A7E"
03076     "C4D415CC0B404B475936F8F675D49C67B875BD8BE2B2D65E05C49EDFE72163CF3F3BCAD87D"
03077     "8E956101964A1C1345F3978E144303FD1AEFA4675F521BBA59F0EF43DE19418993F444C1F7"
03078     "5BEA24B295CDB73A51510B8B4B59DD20829466409444D4FF428B3B94164666A642FC2A187F"
03079     "DD7ADFE75E813BF64DE664B131F58B9CD825A19AC360F4F83E43FFBE0923494D978B1B2ECE"
03080     "EF07F2387E03B1D933008C0131A9ABB08DE1EADEF78291141D285EADFE1228554733CD927E"
03081     "9740D39EC9F539FAB960EA829552CFC36CD8BFD9092C821CE4FB4F29115BF533A56D7663CE"
03082     "9FAF2B9DA374B6E1CE5528A5B11ACB9FBE818BF4C41787F5E4FF4D517D46D2234B77F20ED3"
03083     "928BB38A32BBA350A66540E19B586C47EBB9EBF4E4A671CB41DA55BFBE5B39C025BF8F11A7"
03084     "86D60C9DA6EE5299B17C0135689AC9D880B594EC51BB52398E78AC7DC2439D0C4BE7BBFB7F"
03085     "044661C9CD8D6D7EB16AFB943589C6210882557C65034BD7D044C6E957722C93537473E2EF"
03086     "1AF5A6D05D7404E274D2DA8B83F6D2AFF9EAA55D122F69F469927EB682977715FB1617B363"
03087     "68C22FD6A2A7B10412E9F202C6FC9BA1D7789311D0",  // PRN 36
03088     "18085D480B93B422C6BB24B09E05556F8298FCA307341877435D2D76B9DA1A1932F20D2903"
03089     "A63A0856D9C431FA3C4C591D606E0043B0C6BF27797158839OBCAFFC01E14C6256290E1A01"
03090     "EB7BDC98611EB4E40BF72B04C21526D4139E3E75A13DC25C0691B67F24B195F57923684BAA"
03091     "F94C7E3CFF4830FB5E828B8F7A8692C1A0CFBB9FD3F021D3FCF12C7497021178A2AB8D3256"
03092     "E8606374D6D0E7FE11BB1F313BE2BDD4D977780C1A2053F6CB0F60F89671AE59C178FE3390"
03093     "0FD11B4DEC25C698CE0FD4E451D6A3CEB4E41BDF08F05F556B4394858983D515266F25703C"
03094     "17B48B1AA13056ADB2AF9A865A0FD30BC4BAC0D7B1D185AA2903D1956DDA3D5541873EFE48"
03095     "7E08E83613428B64D2A499BCF3783466079714FDC3D0B04ECDCED5C89F77F5E6E6F1141D47"
03096     "EE7A3C32382A89F512F9C9BD96AB24EBD4F77B911D8B8206CEE29ADC1E055B7BDFFEFF94AA"
03097     "FA750B857EC6FFCA693787C0E2817554717CE5DD5568DB1AE585B9BEED0C6E228FF47C43FD"
03098     "7565D3D5664BEBD73F97DCF73D8EBBA3FA092BF6EF21730EB909327313FE1493BFB4439756"
03099     "7DFA30BDD7D090245057098AF775CE977528E91D9C3976A4D478A5695482E4EBC1FCB2F7FC"
03100     "F6E5F9B08BA8C7DF12E6F77C3B0362B1132F9209AC1D69579D5E9815535FF72820BE141402"
03101     "0694A3E2DE53BBE7DDB582C70455BCD18AA0A4E0A8BBD718217C616129F236487194B266F9"
03102     "5E644D1FA79532D7AC13CD9C839BE14AD4BEC179923EFAC48965A8C915812C0F0886C11ABB"
03103     "0D2DD58FEE6AB2549AF9380B8699F3EBB0F96D295C22B940044819D7FBF2364610893457DD"
03104     "ECED6113D1B14BDDDA8077BB32E70D19733CB795BC8964FD9788ED317B5E433CDFAC3CE3E0"
03105     "EA2A8359671F2ABF49946217358E92828AE32E6CE645C3C0E4D1CB64E5C8532DA7FC179CC7"
03106     "ACA241AE496D821EF7FB9E8D878B746C50DEB281604830360C8ABA336D1E8F444FD46A74F6"
03107     "B1A0F1F61EEC42C1DA2B6D88CD1AE94B3208866BD1757B1ACC09BC155DE4690A97D0244B81"
03108     "9C0A66785EC764276BF39FCD6A1A172C777F2E9A44817D1B6B3C09AEB4436A3B115605A850"
03109     "AA9D11BE2E660F2B214C45B00134604EE395A77BAB321298ADE920BE3050CD2D7BAC4931C3"
03110     "E3687BC4FD7E2EFB536E26404E9950C928A3CFA8A11BE42D15EA8DB2037154F79C67860844"
03111     "D04DF51F5F21E90B8A16E7504947B90B655744758D97886E4860FBDF28FDE80B8F32C94D09"
03112     "F4F827F34E8FCE92B576A8809738D6227C9A31A43CEB3EDE56BF7306DC6F208D6229CC3E5C"
03113     "4B49AC54486ABA7F0079EDDBCA82C3C77F41C58A88C4ECF4DBEA754DFB3424BC3106A9FF28"
03114     "0E8C6A7D086FA06C44ED23A82E23A03D3FD4CF62DE1AAF3A95747CB5CBCEEDA5B1929C13DF"
03115     "6CFDBA6A94A62AD3035C8CBEFA10AD9D37389B2EB90822E10B422BC165A3BA86F37C0A5B96"
03116     "D0EABCAA331FBF2806C65B376665A434A6D59B3BC9C339F4437F4598D0D6C62812A2083C32"
03117     "359B938D78390F9B4F86E42F297405069D350F86B089ADE021C1599FAA2460EF5904A6E257"
03118     "08032540A7747BB254679454A7B38C7491BDCC835033A76514869ADCFE268FE49CF38A9B84"
03119     "4A97214E1E1ACE873FE051FAF282664552DC98618E35DBA4AD9B1FBE50700726091534091C"
03120     "631B54C944C28D97DA0961F8E4106408017CF1EC86456BCC1F4EE7118D0691F0F327A0436D"
03121     "3145E5FBF8FAC33E5179094FDB03D9DA0D9A2EEDDC221A21189B68EE1A94B13F062F21472D"
03122     "B5466B266562E2186FE01D028CD4AB5B6AC9F048D8",  // PRN 37
03123     "D50CD8EB871C5F371648DC07E20BE84263CD676282D56EA5374E21B52752DBE416DB787BCE"
03124     "232226568621ED792A9135A59849451A4A4D8A1DCF3C804AF966881FE5156FB761B3736F42"
03125     "82240FC0BBE72F51F0122D96B3755109E7CDFC70F9DD04C58B4A51C00B4FC0EF252B05ED76"
03126     "440C97DABA04079C121351CD43BCDDD4AA2D2495B49B908A9AF815DEE1405C3CE9CEC7D5BA"
03127     "9919621E8C4E920E08061D228038F73F8E5DD66BAB806D2DD953C6FA5360487957912249DE"
03128     "009655658BE8CACF00E6E0621739648A7A75EA06E93946987E62EDC270B266BBBCEF61D7CC"
03129     "63013AD797589A7CA62B65D4127E701BC95D45C74DBCBFF498D3F87CBD1B288D6012EC392B"
03130     "901A3DDFF16EF1E1685EA1F92B1F32DFED2F9076EF9CB1ADD8B04953E5509B24C48977FFEE"
03131     "6044B8C5B80F9DED6FA7110C3A2EA42EEE8A83F4A28FE78B27E84AAE7B3FABA56C1A4D2414"
03132     "E0341C69C40A43D65E16583A151826F7C7CC0CF0A5999A1016BE6912F067913AEFB525EE41"
03133     "F1B9C82881054E9624C03DECE74976228B119974BC546F823A597C7486DD594F1FEB2B6D03"
03134     "30918936DCBC12E3752A7072C5FD7B145571DC4A22C770E00F24B8C91E27F2C727FB049D04"
03135     "BCA813948D2F7B531C906F2018F51C9EF8B7642C0F1CC06AD338BEDE17683E0BD2A90D4D47"
03136     "46189F792952037E4918D2D6E61C1C8A8B6C437075ABC38913C250BBE6EDDA5CD24B630AB8"
03137     "FE9123B08D5E8FB8B71BC95BDA4AF1A71D5E54BE98B16D7D3083AB1649CBBBF44B3D5DC33B"
03138     "77DC1171A66F6F40E75EB86801EAA36E02D138093F28ECC6DA4CE4B128DAFB8D49A17C683F"
03139     "FAA8E6910B8BB7B832B3A3D0765DE2ECDA9D7BFF0A822F5FD389A04B660F832687CD67F7E1"
03140     "E1C3F5257C6FEB27B763AC7327DE8FDB989413B6D002FDFFB1DE5A80F1C5E39F06E6D716A6"
03141     "9ABEF37B2DA9783FC494574689948DB51F78C8812E09367E6C8D70F5FC4139C64B96C71ED8"
03142     "51D171AFA69D2A646ABA85279B7D3D31D268A2E8A272C91902F12C49D07D74E82728AD61BB"
03143     "ABDA333DE7138A976A2267DCAB1A02E19B0685F64B608E3DA41108D54A07D4BDD24D79D293"
03144     "830C04209C98F037283BAEC00B1923CF404E1F21584F515F9D59B53FA9B4774FA8D3B34B7C"
03145     "66907BA127A095CD8CBB23375CB188CF9DF22CFB4CEC528C581DB63598DDD9A53438409D72"
03146     "507D2CEB5BD45DB5C760415A5E3B3A909EEDA73B7FE122DF04CB1E9F29E7A5A7385F632AEF"
03147     "63771A4CA2164B9B3FAC3C13911DBE9668D1127DB02A94CDCE2C8C7B66F0C21601781552B3"
03148     "84DF0FA002DE08892D0881174CA1F6DE80B98BAE545237C5CEF9E1655F9EF8F8293D644CB4"
03149     "7963734197F1976600CF2767A59FD56B9B67659168ECD57C485E5726C150DEC6C302456AFD"
```

```
03150     "4F12B2F2D1F36B03E100E109827726020BDBA970A6B768C580E34116DD0011299B4DEAAB8B"
03151     "04DE1DCC5FFCCBE57254CA79743D6B00D119CBB2C0B0BBB2FB8FB0B0EC97204E1720F99D16"
03152     "A3257846D09BF8EAE02DB21FCABF03CB0989FB4D99B6D71D5E64C6FCACCABDE63B223A91D1"
03153     "278D676839E1A5CE6C5EE05AB4F10EEC511C366DBFD7AA1E139AF671CBEDB3160A775E5214"
03154     "C862E95C3A7993E508D8F8B7ABD68213A531B5C648C537FA2674346FA0B3AB43A7B58B4F83"
03155     "B36D58C506C188DAF72381DE53F83AD4D9E6C6B1F5D689AE24CD158BB2898F06AA28C8FFAB"
03156     "1703871A6AFE41ABB6B8B9188C90EEF88E7DA3338F119770B31167E3AB541371FF7F98A2D"
03157     "59D3557438DEAD71C96B97F3839F5E28DA6227E5B4",  // PRN 38
03158     "0447B9C85642F3F8BEBD1CC7AB0D0E4D07CAFA93A26C75970C8B40443506AF8BB671F30867"
03159     "FE4225F576026F79010AED56CE915AD9AC001447CDCB7CC3846B0E291DE0E9E443984E0ABF"
03160     "7C1C1D2F6D073B0B77116A21C3C0B7F360439D11D2DE370CFF8FF73FFFBABFE4D52BCE58CB"
03161     "32612870B0EA23CDA21E3B6E851B063BC76FBF7BC394A7D3805D45616C6A313C887D2FDC51"
03162     "F065F8F3C05ABCF70F31EA5746A513D2DB3F4F3C61006F19420AA82107AF855649EA5D90ED"
03163     "0A8540D0A2CEC685D18EC8F4508A7D092E52DB2117C233A7BB542B61C26BB92491A88942A5"
03164     "06ED60354EC1963ACC189A248E37C9367F3F6FE065EDF6A1205BC80B267C1062BE20950617"
03165     "FE0EB59BF8F1E69982F3E89E7EF85F3319A22A22B8BAA491BE76AC7257A2DF631A98477544"
03166     "E3E57C7D63616D22789968CA5E929B507AC394B6772096CC70B69ACDBADF900D055881FDD2"
03167     "09AEF54CE6966767C07EB7A2B40A3410C3417D114FC9E293DA7262171325D8179262B2BF04"
03168     "1F5A977E2D648900295B14950EDF245C7EEB663F8292625F804BB839BE3A5510036F7A5028"
03169     "8C69F94434A03466701BF1313020A4DC4F87D272FB9D1B7676C195C0B7B4D3C2CC6B20D5C2"
03170     "066C7DE5DFB5C9FAD43D80FA43BA917F200A73F258BF21ED8BF3E39F69E501AB3EB115C941"
03171     "4C82C52CE91F6CAEF4A163E288A6FBAD4FA83477909FBC972E2EE2A5379456105E8EA02F92"
03172     "B0D8F79588E2282D1970C3A033F738011BF273EA65912D3ED09398A7F851BA2D6141F97843"
03173     "E90F5C66B38D71D70DB9D93FAEB07FE1AF5783D71CF50931C282CA8B483D1F2E5DC5AD5FF2"
03174     "B06C4355E46768AE5014A22E7CA26267A1E6CBFAB24C8C436535F23D419819D6D458551EE7"
03175     "8AEA4779574BF86DE7960969B0635F0B2996F955D863E078DBF8689C142736BD5177DBFEA3"
03176     "CA8C58871B1FCCBFDFD67BB12AF82C3B8F935CCC507D510CAB63B831721E0ED9C258BA7A3F"
03177     "E245D42DDA1AD8A962847D7C724DD2418F72508FAC194EC43D17533AE522187B44D71B6E99"
03178     "3F2CEFC6349141094C8CF935D90F2F238728D29E1F5AA09B46FA7CB651F688A8A06AB97FD2"
03179     "363486F61D3F42B9C015652A53B3CD66372849FD2EBA05CFB5AA28520B01FCBD9C835F98D1"
03180     "5FE01C84D7FFBBA4B2DB93E44E4341AE95B8AAA4D59CB88F7E192E89954E22B1B8157FB0B8"
03181     "9E3E1E3862BD4C2799F2D8C621064EC9BA1EC30E0F7F5FFD7013A7E1C102DBF01698322992"
03182     "EB05978A5A2878756BEABEDC660DA2D5370A98CA2C0A4C65141EA273EB787B2E7AF646DEF7"
03183     "DA77A93E45C37E69B474D04FBDC3B5982D8492FFC37FDE42385DDFE97A73B8575C1305A0B9"
03184     "AC0B0CE061A423556B6FDADB67D0A5BBA7435929D658FB0869BFFBE5D2FEC96051FE03CB7B"
03185     "0D944304C2D433459A97C43268E43951D4C1708C702FB9C0A0C6D82876057F4D043B21A92F"
03186     "A6D034B1DF5C2B463107AAA814C79022D96D1BD9D33DE5A1FF9E435D9607575856CE4DF5F1"
03187     "E9E90FEF5ACC4AAFBDBE0F982DBFE78C5ABE1D9C85E0452AAD7836D39591411B382BCCE72C"
03188     "A1702A20C9DF384F5BD539F53CEE8B9977AFB705D1D28680D1733BAF5F298282846EFA26B0"
03189     "DD7514422C0EF9088F796B38C15EDF6C8D88009B10CD0D6B4B2FD93679712BEFFC4D1F7A38"
03190     "98AD0CDC61D2FCDC360C66FDFE0AFF0E8A7B26359606CD4F47D28DE115261E6F98F323FBDF"
03191     "3E3046BCC2ACA0831D2BB4DDF1EF443632A229CFE6BBA5179079DC7FA50C56BE7FE5FEA0C5"
03192     "AD09D891895508722861E9EE8568F59616890C6370",  // PRN 39
03193     "8DE87744B81A0E21A62CD72148FC3782AB560CE136DC3A07F2D2E143EA0DD2ADAB9FF8AB39"
03194     "B2A3E8858DA593B6EF6C46BB5CFD252F6DE446D9165406B3DD18CAFCCE3DC7A6E6200D8F75"
03195     "91169A6AB9DB6665FB72140A416EF30B75EBABB6C284E73809DE2A3A26C7F4F188FEE0B401"
03196     "E9EFF6DA22EF3DBE0FAA6DDFFB5D8C265D1A1855CF7019D8659062AD7A705362D1510ED501"
03197     "3B96D6EE803E7C6418B058282434967538E8616F4075A08346038E3F3549C0AB1E761790E81"
03198     "8C2B531F06805D92FE53D45B9A6FA5E30E3D6A2F467AECA07D29E1E9123247C69220E2B9D4"
03199     "501EE42E5BDAD07F0D092B33A938B8FC0EB7E435713E3E428E87DB24AC570E4EF64840A1B4"
03200     "D43C5026C80F321E537755366B16FAFF2423908FB74E9A5B08F0C1064815472AF48240F1C3"
03201     "74F2AAF8ADE55117E0FB1D08AA40C4D8AEF35CE6A91E54A89BBE55BF5E78ED5F66B2FA1E89"
03202     "36342656C63263D4E7ABD70F7899DAC9A315E9508AE65287C6660A7D7F3FF408CD06F3B19E"
03203     "1238E6D5EF040B3E54F4469BEC17198AF3F78F660C6753157603138BC98AA9F01FE79036D4"
03204     "564A7396725E57F875FCDA4EAF80C5E2815862AD52340571A571B331CC8122C13CC58403B2"
03205     "2B61BD404C6D94C36FEF187C712B524BAE9EF150A71CE32366AE536B5B94ECFA351EF0DE77"
03206     "E729C42BBB32C7D35F9A5BB29D2D84BFC91F510F9A1C907540AE3A80CB7023AE0635EA5C2E"
03207     "A0548D9D14CE4BDE142436521ED1637C7B7CF6CC1DA5F826B800AC6FFFB83509A81ACE30FD"
03208     "969495313F9E18CE4D8A2E1D5F9C7DA14A0A9D4C4D49BF00B622B6AA07FEBFF0A8B274C297"
03209     "C0B4AD1CBD64BB4941543FB63D9E15059E0C0FD250753B2AA664A677780C39A013E1AA6B8F"
03210     "786D677755ADA03E51ED55F936FFA1ABD0430DA8750575C37D1EEADEC5E17148DB9FA202F8"
03211     "748B0611EBB5015F1D26C0D810F1AA5A40D73C32C269EBA5CDA134267231FC3783D3D4CC63"
03212     "9E567681275F423439A29118C6A0C1B07D08416537D4707E9CA3FCD3494F64B69E2AE9EDD0"
03213     "079CD428CEB8A00BE0FC9A791FE2DCFC10D7F813315E964828A4DCC2A2D42EB313CCF192F3"
03214     "2AA9A17C984E0D3CF3A0BD86E0B751B30C096F5F0B08A0BC439294D2BE2CB387648F8119D8"
03215     "820D39F17C6AEC976A0B8C89C76D12AFF73059B49EE856B2591E8D2E817DC43793F20B2AFB"
03216     "BD49FD9A05F5B4CB69165420AB96F26A46861AD9423F7DAE8829FD392799EF967E1270563B"
03217     "CF3D46025CDBBAFD15DC23F33EE5B621DF6A12263CC0A506A5E9AB191F896D13382B3BC2D5"
03218     "36442A62B09F3C2C2252D0BE377CFBD59097259ED243EFB36C9AC0AB01B3AADD502DAACCE1"
03219     "7A49CC82FC9ADF67B4EDE81D9355CD8295DE21468FAAA25036B2DB6E24A3AF3E5FE5932486"
03220     "7658EAD5198C47D362EF64B71179D107DA748F00400F15167E84B62588F6D81FA4B68A59A2"
03221     "4BB3D27167D0720718BD24EE556FB72113FCBD37831251DCD538F6815382E119355CF3490D"
03222     "F0AD552ECAE0D00462C10CBA4011E95C7FEC968AF6E39E1FD15D1026879202C57E2CF7AB02"
03223     "B08B15373C13435D8A55ED6B9BDD98FCD5B4539428D90A2C73D37DB112D2025EC8A4AC8EB8"
03224     "2C51C46FACE88AC0E1F161705801C781EA07F6914122E5DD6BFFA3812EE44E314BC09B785B"
03225     "344584AA8522B63FB34BAD2122F9FEE1245E6DC837DB032387EFF5036A021C8112CDF03BEE"
03226     "FD89EEFE1BB88A132CAE9E1E9EED3A855B364F2A4A2F81086172FEBC64614BC04D11E74AF0"
03227     "0F54AF6B5D85EE4773644C3FA768532B1821EC6E5C",  // PRN 40
03228     "0D1FA0561DE4DECF411AB73FF48D0810AA2149FE5F3C22E62E06C02F60189AB5690A991CB8"
03229     "8DFA5C4FBED745FCFD63BE3ECD9CC599E35B5FA31AA11C62F3A33796A64EDD0B64F2E51E75"
03230     "C2481D3EC9DDF07DADB71448BB336C0DA9DBCD897F777E3C9DD97C7EBB08827C19316F6142"
03231     "0D96F3D94EC395F8BF88F5927C71397F6C24901914826B1BE26E14FE93EED37141109CCD00"
03232     "C92772B96D9EFD74B3EAD3903854801B22DC3F98232023E1F9CDFD4B4C952A6D268DF9FD3F"
03233     "A83C707941F8F1ACDC195AD66E7AF48D7B06C62F3204E8D03D2C92920C589CC56A5C2B0A1F"
03234     "10401BBF747B60B9D2179BF9CD7DDF50B10823372CC4E24005D73B4C5AB918F22918B16D98"
03235     "F15C365FE69786C1ADE9FAD71516E389938D5D38420FC79F82C6C0AB1C6D93D1896FE8A2BE"
03236     "BA134C138EB1FB00115E50A8A676B9D9939E7375F4B7D62D449EB341B0C03EE3FB18564DA3"
```

```
03237     "CF64261055594E0F8D322EAA9B56B5328574AB323C4376464F3C27786DF7026BFD6C7757BA"
03238     "EDE7887217A2D3EA22AF7D6809BFE4985EDB32ED503C6034FECE55F4379920C73A515C7390"
03239     "B3405ABCC54F51F686903D5354468AC566D5A03C5AB536934A6CE698E06B9E2C815B15B4F8"
03240     "A1857FE7C0B5486759B150F48AE9745AFBCE856830078EFD7F4CB4A463587B7DDACC79FC9F"
03241     "BD9C2A0351D91DDF27A65CC79005DD24F1C26252BEB8C3F1D69DF79D386401EC5AD57BF212"
03242     "9D1E828D6D57CABEA84C7F77B16B919653C12AA7F770741E4AC11F0AD08507C3E7C4716F14"
03243     "F14C548A88DC4047EA96CF96BC0CF786EF7D02424314BC846682109A2F80044B51B12A0350"
03244     "DB9FE06E72581EE874665B9680EC118C86DB575F9E5F687CF35DB3CA83475051AFDABDC974"
03245     "B2991B47BEB4CB967E3CF20D2B7CA2BB50825EE33FBD7C6E87095DE8E36CA6143F673A8C18"
03246     "882D89F22C08971CC709F1518169123EE59EF50846C95C313F14F9472D4CCB8EB713014889"
03247     "01F27E895B5D44420F91D123E3EF5527121B9B0C9536CC3D4E5EA03F63BA8FED0FDF593AAB"
03248     "C3E768962DA165CCACE066ED7D6E46DAE97EA08789C36BD3CA6888229D15714302A39EE4F3"
03249     "9DB0751F0E99877237524ED8EAC5F7EE2CC40A69408DC43E2AD88A661D7E443D7E99A14484"
03250     "E3011A41912E2AD4F6B7D62D8D9F3332F79A5FE1E16E8064C91DE56B1F89A9A3A220165872"
03251     "128771745D086DD3F8DDB2C35AAF3DC6DA70684DF270ED50D188FAE62CE4C98B9019D30877"
03252     "2EF036F9FB2775137377CE61529A40E03B388CFDA3BAD55FD62D7ECAC2A72FB68068BE0838"
03253     "3CC36FB66CC252A0412CFB3C993803C038A1828E8893DB453D47C5727CF8BF80850346063A"
03254     "F4976610F984ACCA00BB5963D5DFD018E0790F4A6B14A9B5FB9517783EC1F16773C49A744D"
03255     "8576F7957D4CD4C4E35D1BD580FC76E2632D47661A6838344CCF6476BB793FB12F92143524"
03256     "E5A01DD5294B6528C5A18B24B0B75D017C4DB3663BCD561AA8F27E510A7C031ABD3708385B"
03257     "03294E3BD1A695C573701B9D4F60232DF767439E252B7DB10D4E3FACAC8C2E8D16F9500828"
03258     "826EA807C2C7C48C0EE8457E757854C1540EFE9BDCE7C09AF5BE09AEA54A4BB0D2A328092F"
03259     "3B8741A128D4BF588384A3848A5E4F55DACFA64F7FE081366330D16589991C52A1BB782364"
03260     "AE969E8A21260518662B00EDB117278170CA2C0D186B173F93879F6F702E03DE768CFEA1E3"
03261     "9FF8453C7B3AB46E83FC90FC56D0EC0847DF16A4BF1ECBFE29D94CCBD49DABDCC5FB81B62A"
03262     "2CDE7D2DA3E67042944F33A007096DE5C79133C7F4",  // PRN 41
03263     "C9FCF7D2DCE0A2341FBEAAFA64F6A50A93F3FAE3B55EB70ABF51A4C001E15A8C78DE54E50C"
03264     "CAEFF8C2421079F7333541018D5EB8045C3AE0F0489D13C593EED560CDACDAEA39BB49FE2F"
03265     "D9C956197967830A0886182D98B5051080DE887638F2E3E626B8941A58D2206B46FB410C15"
03266     "5B94FD7D15956EAEC9F855B96A8F1C7202EE600C12DBAFC6E3DE5F075B05AFC037847BB521"
03267     "D1D3638135591B6E1A0137CA74E27ADA3FD673684A8A1330DFA46D449F8B63FF90E36881C0"
03268     "9FAEDF61672AC32CBBF71F34367E7547C13BF08C4B688A397DD4D00D8B19212F85EF11C0FE"
03269     "A19C890D73DCAE9B532CAB7BB5931B2F648072F35FAA31C6A77FAF3F72F9F57D5AE65BBC7D"
03270     "FF1C2A5A2A2721AD5B50DBDA2CEBA27A3864A6628F24B0D24FC2450890E278C4599C1015C2"
03271     "B4AE61F2DB2A3A85F02882DCB57A7A471B5F8E51DDC284E0C1869D41EF25E260EDC53AF55B"
03272     "E8217FEADE6C74B9C0EAE7C3417156D671577E05A63E03052663F420B70E99DE068971C5CB"
03273     "8D9055864460D6B2C269E4E5D2585BB7A08E28B15851C6E8438EBC1272D517A076C08631A9"
03274     "890F4FB627D05B91ED2164355D7F20B32498E4773FFFD17F3B15E7F2BAF5EBDC98E617A743"
03275     "7BCF94B0EA1B2FD6F98F8BE6D64F769FEA28A5FED9DBF4B3700BFA8C0495772A14254D7AD9"
03276     "525B6ECDC028C43EEEBF91D2B17C9ADDC071CCC83EFC6401A22846500E0CCD3D4E5184A05A"
03277     "ECBD92808AD2DA42C57621C66142784692D4F28662A2B37DA4D3CAE332A97D465D3E492ACC"
03278     "964AED603058CB6BA6972C83C2906CFCC813C0603B775E2DE8BC946DAFA510532CEFF2B2FC"
03279     "5E1F25540FA07B577003D67F8A3C63E9BC2B89EC7AEDD8C2E91CA426398BEC4D98FB623383"
03280     "2BCAB8F10E5DE6668F7DF8FE10AC0EA63B7D46837EE4D1D203D546150422035524 30DB97C2"
03281     "4FE0088334BA154FB2157DD2D12715ABC6D329A819A1A664727F40CC26204949926E78B0BB"
03282     "8947FDA9026813FCF19795F8BFF2F56DB70DB7EA9280A6C504134A1B653D336101A9C16688"
03283     "3599477DB557406FAD495BA31B488D9EDEB783759EDA32968AE4B4B96A14C9DA960ECEF511"
03284     "D4102E1EC27E17AF2746DFEA54F42D2133281ECA9F5CA87F8A2DE4C937B5F9526591A583DD"
03285     "3211F78871BBEDDE79206F946B9CEC6EBD0DA4A25AF259E4B3EC2195578C74B7DF87963BF5"
03286     "61D7F4C6755D4DB0F288341D5E525B15F2BDA314DA38D2B2C7CD7D64F6DE35EE782A076333"
03287     "941FC3505AEB1FA5783C6C2553D73AFD6AF896A5C2E1C15EDDC408B6D9D7EE2E7E9F706C2A"
03288     "6CDE36A41BFF88960612C46996A9E7D43511302F925351240B5B77A7183C45A1B66C8FCD34"
03289     "EB6EBF5F5FFEAC8441BD1103D2151689BFCF8D4E29B9C39B1BE4972186B099C2C35572DB1B"
03290     "E2A28D751D6857B7E4EF181B786C2A6FFA30A5D6E2E1120E7B1F313302ADD1012E43EC24E5"
03291     "E5B5D03950E009272452ADEB9DB626FF38F335A520B12CD981636BCFB882FC06B440E34851"
03292     "2E6ACF980A1ECF360E33DCADFFF5A2B413DD214D5AD1070E88B63FC97652 95BBC92C557F3A"
03293     "AAB6AEB293249761F95EE4338243D35AF883AC33832155AC758ED05AA1BFFF0E9FD58E0338"
03294     "4704DA5A9237F1AAF52E12C821BB0F54C0820AF8124AE6D0F4BFECDA6500995B2B61ECB062"
03295     "8BFE2A9E6A05DC94A781137522E85A3BEF28B76E49B27BD05C113C9B3A9AB1285BFD926D65"
03296     "BCCD5CC3837490B21527A1E94544C4ACED0352BC84D378625E3655F74B039CD681D98579A1"
03297     "90AB2B89104C83F46235757D2D22FEEF59A250B564",  // PRN 42
03298     "48116D6CCE73D8BD1E1F45D9676B031D9B0E65684FB88A83D791CE8F5278D33EA9D3A6ADF0"
03299     "4F29AE93123376D1CEF52F27E875EC0024A888E688EE2A21A4AAD0448103569C8A7CB2FB07"
03300     "2E3D0D09115AD8A1C234F7765E77042156AACC4D4A8AC697E41B71F436ABA09C91E9E4F2C5"
03301     "B573CB1D0B291F915BF922FC75E6FC6AE332B42D7E51BA8EFB1CCA66EC849099834E48C7F7"
03302     "9BC16C79B0C87A8EAC8F88D30EEE8FC0EEF8C89520E2602A7226BFAC0E8B30A117B7188756"
03303     "87F81A6125D09EB3BD2B2F5209C386DEC5AFF7DA0FF5454FF0451F825E5AC55680EC7DEC56"
03304     "E72CAB953E5B822E0A1D95F3F948E95DA7078C850C43AF46B4EFEEAD6AF807CCFEB1B61CFC"
03305     "EFB770D1923B41BB354C81F340CFD851EDFD537C373CB9F29303A988635C5CD37AFA576D6E"
03306     "8A8C2B7C56DE2FD49E5DB970B50E04436B1F8B22A6DCEE3BC27EE1885CEEA23E12541554EB"
03307     "2EFE615A140831DE6C555804432CAC0A73DC2EE5CEC8ED139ABCA4D17B22BCBE60F05F5D7F"
03308     "526ECB29A51ED6ABB4DE4A2865AD1AEA64041EB0F505BE3CB8F351BA21A8D941902BA56503"
03309     "9EFEEAFB87DC6DECEC1C091ACE228931A147C491C11169AC87884D5EE9DF128C98A084CC6E"
03310     "A0A50D494FF8E0B902A9EAA4B4F9D3BB2BA1FFD3DCC93C45133B9008E2B188F57A82B5DC01"
03311     "862D35F633CD6CC3FC12BD6F3A20C1EEE1F9018E41027788950DF09C4B482837677B33B15C"
03312     "E24EFE6797291C0DE291AC04F04AF975703BEFD3CB8E59C1496E547285E3F7A29F99C5F210"
03313     "327CB328B8A3094DB7C27CF7FA077791160A2287F44F92BA6F5346E48D4D4AE6C2C7185926"
03314     "526719537AAD530EEB3B0B4CD718C7BA38BDBA21671ECCE0AB4044B797B7D10797EB1BAA02"
03315     "4B76D0B82BF25C946555545227B46DB2A038F87301BCBECE0C99A98AA5BCB648DE58B6CD59A"
03316     "C6471462BF5985C595B00A4DDF2948A84203890E2989D8DFC9864392E7E75262DF1CB78C1E"
03317     "2872CFCD22D561F8788F42A0323CDF3EE1588EFF2F7BEDACF5AF710CC7C6F9957D992A7FE3"
03318     "9DB696DA414AC36CBDC99AF050023597734D94C2A8FCBF41D6004C21E4F4357C19ED094887"
03319     "97CA012F4BEF8F1743E396236E708D96A7DAAB07FE03D5483319241CB35F66F66D99D981DB"
03320     "4427C4C84A9EA24AEE4F7C4E023E0ED2FEDF59FEB6D8AC0FE87DD42580F7B0CC6339752230"
03321     "A69BE83AE13FC12AD4719D45CAEAC6E8695F6F403A0713ED37DC120B80DE82F7AC4991B4B9"
03322     "DADB31A304E16D03251CA140659CBF99E891DCBB764293A967DE1E96848B8F225D2F9D28EC"
03323     "2F428AFEBFB57AB79B126E6FC3D66102E3813347D7CDA5499F6AA5BA6AB5EA6F3A82D730ED"
```

```
03324     "F3FF74B9BC0F954BED21E77146A5591941962C49F6B2BEBD060A9C82D8FC780FB3AED527D4"
03325     "0CC426988F042841250B4A85A3F3501FA3436103313C3C4CDEA772B6C840DB866B9730798B"
03326     "69D6AE12D24E4DDF31FA9B4C8DECF6CBCEA2E8A0F9CC67CCDCB64523725972D81ED2E65622"
03327     "83DE99297458452B94467B83F3111CBFF61006814CCB6550D19A1EC026996695D46C31BFA9"
03328     "BD371CCDF1B512C61CE7BDFBE419B477DAB1E7E1D012DA4BDB05F4C4CD39511D5071C0982F"
03329     "68D9C6F558D22940D541A3C67003D05306FFC070C2131832F737872114566ADF3074F7F627"
03330     "4AEF9B894D0AAD5ED60998A0363B92D841F4A441C39965FC1479E93C99B083CDE6CD6CB9E2"
03331     "6D49E16B969A357E28A61B444AB0FA665B853FC89B64E02BB02C5273CC948B000417014594"
03332     "08C95E4BE49EDC7CB605BD2DF31C3C8BB2AA9FC780",  // PRN 43
03333     "840BCC55474C2F66B7E16C6F527695D367F70F570B13C1ACB546D8439A1AE06F362C224FBD"
03334     "86441D82430A345B8458EB666BC93739E3D679754117C95100E8F00879B891C9E03788543C"
03335     "415AA964C3F861916BD7C141777A93B374943CC4CEDF928AEA3EB72F412DD8256227D7D724"
03336     "4E876955B13FB2EAD1C3980AF6F8F36B3E71FE3B189F8DC22C55E3720DA11C6AC4328DCF67"
03337     "0CF948123FC7280D6891EEB356C195D2A79326299A684F5D68DC0ACF00B780CDCEFCD7CBE4"
03338     "140FB5B6DF896883D3C6FFC42BAA41E6A75713F8861C18A833C688C2BF6EED7F0CE66242B8"
03339     "196C7C254F9D48FD56AE5ADA9105A5C4AF28D0042E059A90979C6439BF6D498991CF64C680"
03340     "80B9B5392CC9FCA4397647A9AAC42C3810FA33BCF31C89A641D1078A0822D93EB773F418B9"
03341     "B3B20D5FEEE1067766AC561525F88F8F097BED0CBFF5BC498CDFC9A5B9D20F4D4A03CA1905"
03342     "63CE543B2EAFD8DC6E4A5885DEA120DA7D157D00EB1329E500D41C7E6BD603E95BFAE227A9"
03343     "B613A71C7BEFC8449B668C59147E1DD5411CBD967CBBD46A9DDD76C547506A54A8245532FF"
03344     "9AA5312607F1AB99F4F9BCF85D93B58D76C5583856595AF5CA1AAC07C945400B3588507180"
03345     "2620855E18E0A3B1EE92BB95241EFCAD82B3435BBB8916C4866471BD5715882F6BE508C764"
03346     "8A59B2AA91A69F60C78482197C04B31759054FFDAA254A143021E37874DE407690B9ECF8DB"
03347     "99CC16B004116F51DC7853370BD3538E14A0F02E7BF74D66FF898AEE93BF0697EB6E8E62B9"
03348     "5C6C525A03E1A89CC2F56BAC591FA81BF6A1B3B9F1259F13AB45F2E9B0D93DA7B4984A9C71"
03349     "272CA261CC4EE4A44FBA4CD03C75D216672C18A60183A52B28796B356EC498F4926D833D94"
03350     "007E92EC190E9713354C7506BEA6B7B5DB2BE659E2A5BA92D00C01611347F67971478F2C71"
03351     "648E33F99EF201367CA3AC8814DE0F83B0BAFF34A192349C5DE87453167820C0EB0AC8349E"
03352     "CA683111776980FEBDC993314278B37B289B528DB319CF59B09B83342C71FD5E9F13EDF4F1"
03353     "8783FF328BC67A03F1F62FD1A4631AFD76363DB1A79CD1EA95FD43934955AE211E00EF5AFC"
03354     "114BE00EAE2A7AB63042992E086810E61AEE3CE5CB0FE75C04969485899D552AD4B006D543"
03355     "BEB93C1E8578168C66667D62CBCB56F98DFBA248EECD891F9936382087F2AAD6F83FFE25E7"
03356     "7F8BAB9C62675D4024E9BDCA5581126BD1ED6035994029BB54615E593D4CA81B31A5AE8CBE"
03357     "F8E1A81C32698FB7B4A339A7E4E575460F79C4F4C5F57F9E8DB459177067869E03E2E16810"
03358     "04EBBF62B4F60A9EFFE47921AD22C7236F25438AF46885E120825509D1E28652361EAC64AF"
03359     "1806317416E5550A39AA8DAE2C6B970D44F4F38D571F67E304AA57BA435C4E06CC97C84842"
03360     "2216BA778B2736C30200527960A66D63280AF7DC691F9D0540AA47F0D26416B46A6D53B0B9"
03361     "3F1B7C3D4DA8BF8AD2FE410ADDE00B0372E9E830C09B206A08D968AF2F716DD04D220E619F"
03362     "DE28FE70286E7EB9C2068F7A37413BD9E65F1B59C331D45AE914731A564D33875A0939A218"
03363     "5348C197CD14DAC59C5E2BEF09876353FB16D849EE67A774B05C51AA2480F18DE40051654D"
03364     "D2C7B9E3B753FEE5696B10BE40BFFE89487DA1E8B1C2000D53B2A98B4DE6C9DAF07F0B4DB7"
03365     "2834361E520E45F5E958FE0760F229C16CCFE0BDBF07E186B28AA69231DDA8378DD6338A94"
03366     "D23541B9D8B63B85962C6304E1ACE2B56A3E433F7E9B2A88FBAD5FC71703231D38DFA8C378"
03367     "E15B6503A3B9959E1759EECCBFF14AA09E6BB8F2CC",  // PRN 44
03368     "152004D3334E877FE21F773408369872718CDEE15BD489D97F606779A9A38398524E656026"
03369     "0980FABCF179ED91D699D52592AFB3269FFB8DED36AFE7DC79DB773DBA560E3ADF8E12FB8A"
03370     "08EB56BFC4692BEA05AA288012B580E43BE085CE583E9C9DC18A9D32567EA950D8165A2F8A"
03371     "443E7AB5E79F687690A8D8D92FBF877C5B84D26E6FCB3671073BD808A7E908130A845C95A2"
03372     "F5E360645611B4B55B03169CC1F50AE2FFFAAF50FED8CB782BF3C67775B613CF82A2D10219"
03373     "9E49A329496C0FA24AAF6370A3636EAFEB992C663C308AD181DC308D81FFD7135059163007"
03374     "32FB1F8DCE238CC4104D26C588F28FE7EE4F2B8C2024357D52C7AA29D339249086CB2202DC"
03375     "1C847210EAEBD3F06A39642711B4608638564B0DD2BFDF38CD79CC1CFCE1226CA835E6EE19"
03376     "E4089A6F18E7ECC3597CAD3FFFE13A5F77A58C2B079DA25B928396FFAEE81FBD9C6243CE0E"
03377     "B0872A187E6CF99F16B1511672821D11EF88884964BAC6686C2A7732035493FB9765B7A51E"
03378     "845C1C79FABD99438FD84C09642FBC7CF1740DEB012AD050F5E8FF03E859E85C51D2E87AC4"
03379     "1185C67C33820EFFCF2D79D384B11B44A806ABB8247AF36E3972222C805EB9854D9FCCDC7E"
03380     "58A101A5239E60CF4836C38A6ADE67686E6487547933859D46D046BFC15AE95B8D0A42A57C"
03381     "401EBE78ABF495EB390B8922C97270C790C2FAB9849ADE48EA8F4F7C6996F6DB661500199B"
03382     "CFE1EAFA1ECCFAB1F674E0BEB1D03319E73126D7191A64A80FCBBB06B1CFB718275BDDB571"
03383     "E8F53FAF81DBBC0522263FCDFB537B3BE9078F2AD449423152BB9DCCEEE97003D1988540FA"
03384     "E39CAB62593A9EC909489415EEF4229358229BDDD45CC35EBDBA088C4BA097BB0E28814BF5"
03385     "048F4F64E6A587B0BDAAA8A1B3E3E69D3D7B6782FC42474773C61DE12B4C6F183F9E625E33"
03386     "7EFDC97A79A5D2C0C939FA668F440688F7A3E0E28A33D36D331069294DBA8BF768010AD93A"
03387     "224E28964A780304F59B7994D97283272824993 4B5347580C16BCDBE9F84EB34EA7F764563"
03388     "5437B0F5422E3DA99EC2D53B22B1A55BC783CE86CC6B1BE7B11F837CF2BD09A514B12ED7CF"
03389     "0AFC2F70F1D6D08288524263B1129135D664A48B0596C4E880E4E01BA3889014C07ED1B7F1"
03390     "72AB4F4A69FC8C04F0ACE8AD1329A4FB594E9EA30EFEF31BBE418CB5515CFF8A375CBE5851"
03391     "5B32E799E8449101093A053EC99F9CF78122FA2D36DDFBD258EC81D7B4D18BF882663CBF24"
03392     "D2A35A93C00AE8EA81A3B0E8040C1E259515156D67DC76602924BA51507C4994221A3B1CAD"
03393     "674BECB11574212EBCCFDCAF6AFEE288BCE11B7FFF1D7191AB329B04A237F75B2046346769"
03394     "32FB0842E2AB888E7D5A4A6B7F77EA04ABF4B9A567E5DA5267F35CBBC928B5F607E9C08359"
03395     "F0031A934152D77E65C937181C92EF4CF17F6EA45171DEBAB54575579509776631 8CB132AC"
03396     "D96CDD65777F41BA127CF251A4E9B3348927757A42AD8FA83F6CE7342D935CB54984E45886"
03397     "D888627F0228E50082B85E8E9DDBECAAF049C25C4D7B2EA919716271500D81EDF574C5AE91"
03398     "E5F0FAD7585EF69C5051473AF913F887B31BD730F67B4A44081BBD567B57C061AF46523737"
03399     "4E0BF4753F3C6CA28D45220CCA3EE6F07F93A29982EB5AAD70763ACBFBE55015D1C52B6495"
03400     "4D4855ABCC319DDCAEE75CDFE3B3C31C329A77C76131F18F3C3904B3EA4E691C03C32CA5E7"
03401     "A1F9460760231473115B5AB7E159013C96AE4885566065883CA3B9661C6773B77819BC6875"
03402     "5D529FCE05E80E08A5097EB37BDDC1AF9BDFCD0A04",  // PRN 45
03403     "0D4897E7FB496EF0C6D62C034088E7E9B8DB52F232CDC9EA88429655ABE366C54CD534DC90"
03404     "38F45AC0D0362E31086CA2FC0FCAC3FE71B2D4EEE548F69CE4433601CD3431D01BA6F0C23D"
03405     "612807EB8E50BFCA736D917F67DD72631BEE8728006D2AE69D1A1BAEA8DE50852CEBF7491F"
03406     "FF989C37A54EAE31652230A0D3E4397B051222C4A22BF481BC52E2012188C7576A7ACE8118"
03407     "5C85A2E841119B47B5429DF277A976F15D5652156445905EC797729A6647E1B593484ACF27"
03408     "24E81786A0C062DE87759E82202A561FD077D64FDD3226A1E290F660722040EC7AD4383431"
03409     "E195412DBAD0AE620BFE808D65D3937AC9C5AEFD6FA106C88A6B192D4FEC1FB1A06907B079"
03410     "6F724AAC6A5EFEBF50C168F990BDBD6B1D23E098858C07E8D2144532AFE040E45B25AD9AF7"
```

```
03411     "D92F9DCCBFF3DF25DF574A4B65A219289407EB63FF7152BF1E651C6F124BE32545B50E8E2C"
03412     "E37F1042A68A62AB70D9DB540D2B39293E008EBAA632B85EF1052FDA17A07B9B1D89A76FF7"
03413     "E7EDEB7090496855B29A9174780D504F7228F8B7436F4676F51975BCA614554FE6B3F946F5"
03414     "E0D1016371F9CEED4521477CE8E9CD0668643686D8482CEC0BB4D6A9E8678B1835F0952E27"
03415     "00FE0C85A8683DA91638D81A85231CA4C7B55C0C0DA57DE5E6553F22290CE7EE72867285D0"
03416     "96C0FC16A60A664C6BAB048939FF9DA476024235DDDBF7F264C90ED44E953C4AC0FB50362F"
03417     "B2CB4C4FC2097F07B71772000D2C2A5DBDF0B6FB83B1BBD70CE9D6A8BF1F924AAC4DAEB6AA"
03418     "DD891A8974C9BF46FC6038F6B77D6AC0D6107AE6AF5B93B186253AFFFD8CD21835097C4B76"
03419     "4F3E0B2447776549D5045E4346FCD3B7880F1B3BA794ADD6B7F291D3534DF034F822D0A41D"
03420     "B2AF9F1E51F212D5631D2525B1804A75B50D8372B6A1C200B81AC1122AA3C13D7463C383FD"
03421     "123DF4029BDAF6C227B3C4B54F75C5315FE139F6A0A3E2E819D3062B3309E2630EE4A64D1E"
03422     "3DBF23B489B96E198BA98D8D3347217A5FDE4DE453012E8564BAEFBE3939E59C0A725A75CE"
03423     "DE373501F9103011FE0EAA40C583AC73C92F47BD528600C42E0029FB3A4F03A3CCE87E5AAF"
03424     "4FCCB47EA8BC62E0F542DBD455F7163A2A8803583896E3C803C0D01846EBD8BB1B3FD396A7"
03425     "C7BC3F72A9D0503C44FE30E5E77E0FC8B373EE931CBF4A9867406BC91972B946B9ADE62415"
03426     "CE2FFC71C2E64DEB7E6205D635977C778BC40C11E36BFC22A17FE641346C74DADD255D329A"
03427     "EFC2CE5B4AC5B195FE2DB453F1B41B3334D5C480BBF10FDE88C722091ABC85667F2B52A350"
03428     "5A142E4D1546CC41E154B345E8456B9CD0E241D70EB4876A70623DFF3FF43A8D66EF60488A"
03429     "7860485A8E518AF4A7056CD02C72E1BAEA3944781642A36932ABFAEE3B88B57B213F9B358C"
03430     "C3F05154DF8D2132A74DF27556DB6C5E45256481C09D18DA604CB91E6AB7AA29C0C1009A40"
03431     "6F1B3AAE268564E04B07E4E9128FF02513AB8A8665BFE1E352F46DBDE7870EE027A6E84560"
03432     "014659888EB6AACF993D7C9F9FBC2974CAE290059802F5D224520CE445C404A2ECC57CE6FD"
03433     "5961EB20DC12D436F0B93614CFC844B476D449416F54459BAD09F886CE3C1C307A91E62FC7"
03434     "36051E6F6F8F5AD3372398458299EFDC1809997FAAAFDE14B15566B381714E3522F629A5ED"
03435     "9DDA1EF4B9D4498DD55F8B0A48ABB2F9B1FCBD991B5465108147BB6125FFEA6B6D479DB6A9"
03436     "983A51C041375B39FE6546AB5B5E089DD7E228D5AB17127A3A1E3AE82A31E952E660F72A1F"
03437     "399ADEB4E1E11C390BB8652F2F7B6D05E9F4B6FC90",  // PRN 46
03438     "AF6D254288BF564C4663014C7A70EB3739D28F1F17B8627FDF714E85B48EEFEB1FD4565463"
03439     "28BF2E2EED112992AF78911A89E1248AEBF1F79906EA91D41CAFC6E541D2BB40A7050C5AE4"
03440     "B66547C9DDA88590AB8B0A4B01B5F8216447A310D1E52C155F92659835F1086D381F1F2995"
03441     "138856DEC6FF59380FD6C1B6EEA6E4061930CEF7217701DDFE17742721794DE824377392AF"
03442     "3D949ED9B0455C58F7395DE0FDA5485E310E8457A4270D603BA5AD1854811530FB7178525F"
03443     "4BB1E5DFF89F6FFE0C83C3E47E244D8F3F8AAD8374E3FDF996060E18D2A539719282368 1CE"
03444     "6DA561816C0EEC0C0600732A37731ED13948439E8961BE44D32A3097C1AA5E19876ACD21DF"
03445     "8122651817745 80A8C04ED4FD1DE2FD5A1DFF9DCDF4D97C448BF9A4DACC22D4A55F588C87C"
03446     "CF5C9E178A8FFA032A454908BBE41E13D829FA98C066A7736577CDEFC6D5441656D0AC95C8"
03447     "72861BCE0A28C2ED6E9420D4D167F37F96087F9E761F38B882CD41521DE8E2C40564C0E001"
03448     "EDBC04982669B82F05CD0411A6BADDCB704B545FCB19A286BFFAF0FB6DC889BCE75C5763BE"
03449     "1D70DCAD1D39F395A75E637E0663F3D5A855F53F700AF5F82124E72CC2A4E41A4B43FCBA41"
03450     "E55298A1B799368BD5F9B507FB11E8BD2D80FE517A5FA9350006064278CDDC5EED43B7C2D30"
03451     "9BECCCBB2C0AE3B8D09509315933E2DF91304B1DD3777EC9FA2415F7D18A96CD08095A30F7"
03452     "EB9BC12F8739E409EE7DC062703CB3EA591531F47F5D1C0431E9F1883C3BA249492EE61971"
03453     "9C8B3BDFC1DECF7E3C4CE4CE032FB2882247D1D54566E3819464837C248D35647D273A87F8"
03454     "BE667760D26A7EA1FFA682A931A4D35F7407FBBBAD725BEA62449623AADE97A2789F2F8642"
03455     "F1CE12A06C93861DE9D27C5ACE0E4ECE40B58664176113ED6437B8AC8B4059653EA30F71A0"
03456     "23544BFFA685343E278129DA12EEAADBDD5AB0E2CCA0B840E87AF0D0E859449893276B9B58"
03457     "94EB0C57C3222E01E7538FF7E91807C10CDB465A12CCD358151D96856C6143C9F8D25AF8E8"
03458     "086C83D4CAE37BD660D15C0C0707691E8366C4AFE6DC0AE1BFECEF9BEF46CFEA3356A3155E"
03459     "027D1C921ECB852BA4B8FE8E0C8C00829CEE4049A21A1C3424D2BE532B735232D12B81C3C9"
03460     "E5B02294701E6BCBC8607F2AFCFB2DD9B1142B4D261418C1B239B0B22451D7D3AC64636B0C"
03461     "7F96B41D7E775C1CFA2E277C957F0D59BD1FB02226A462533850E99E917E310A684E81634F"
03462     "3876D6AB32CB5B8FAE9E82224BD2A17A7C72CDACDC6B3438A083A6B0DB9FD91E05BF4C83A3"
03463     "6916EF35C5D4871E94B1671493FDBC497BFA30328DD24D340A07A960828A855FA95EE33FE4"
03464     "FCD15B14B7929D338611239AC34CFDA6503E4DD9570EC339CFDC8D1C3D3536F5176901D012"
03465     "BABB133EBB4560484344E044F86A77C21183C82BB2B6632CF69F5A661C98EFDE5529F8FB96"
03466     "5DCC6AF185CA8CDB4D01C5E8902966B063B8D1724AD18DAE495AD47416B61321DEEBF782EF"
03467     "7A6ADE3CD0D5B9ECA84DB5D50C2E5C7D8B8DB9E0F398376CDE3D7E8DC46F0D77D38D6CF605"
03468     "5D6D074CBA92A27C6F9018CAE87C97F1B3ACCEE495ED9DCE46DF4FA5255DB2C7F2669F5201"
03469     "058E90F7567672162E17D5B6D028C6AD23F2AEBA32061E6B33EA86915FAFA2344CC58162C0"
03470     "32CE837D5C32F5993013314BD261DE6EBB35B94B8F690C5624A8CF5509BF50D3AF5FDAD773"
03471     "11B6D6F9398213AC1A267CAA0F52A262DA4F85F5ADC79947D2C5128B8E0B0C41B9FDEEF6AE"
03472     "C228EF97762CDBDB78273C16C39E446576D54A4FD0",  // PRN 47
03473     "4B7593A9ED7897766CACD515D3A55959FF99C1CE28A745DEC1D8F2565F24F4A1E14E9083AF"
03474     "F510F106D982A2911197338499CBE38CDF3D99463B13EE26B47D26A62CE45EAAA04A3E7085"
03475     "0D5F23470FB94C42D3235D5FF9E6C37CEE8C93591CB69E0735B03EB262CE6BE6F0144DFD66"
03476     "BC089B36D66287EA588C78E39D9B6907EFE85C1211612952CF13C369C2AB3D921E4630ACC7"
03477     "5F2AE99014776B26CD1F296F736A4616FF662D5C6E18C4EDA6D1791A71BE969556FF11E119"
03478     "2D3941F8020D2C731403ACD856A3AEE6ED7F23023BAD7BF138C702B6449E2601042D7990AD"
03479     "B988B650AE202F3433CF26EE132A7CB13650E86A6DBA6F7FA53B2354DEBF1268734D7120F7"
03480     "21E18FEED2F93C268A2D3EA012F3D7F68DB0B18A5CDEBE13CE4A05683947DD985D4AED1E19"
03481     "2FBD2719755846C9B758FC8FF28B9999D07E634645064C2C9DB4CFE50BB8A030B60F43AFBD"
03482     "588EC17102F614D3029FA811457568D7726C651C062391A2EC2843A95A3A48AF58A898BB65"
03483     "FB852F73E9A82C6AA9D406D80C072A3B426D8EBF261BF7AE0E9DC0C6DE9F4BEE880D775783"
03484     "F910AF19DFD8EC2656213FB9B74EEBD8E1BF860E4650D446683B7794086ECE1E2AA723024C"
03485     "219E3DCB371624C6D721BB60C797003D89096BA0F489D1CA60C57AD907BFC8E97F4E057B6D"
03486     "709414FB0A302D3057FB4635F70BB6A32CFD842DCD8D9C9D45FE082B3746951A862EA870D9"
03487     "03382C138425DF936A505120D93FABB8F523C1D3946B85425FB338CA7DC4B2FB6512F0C8A6"
03488     "7FA47A416284EAA943E1A9C0607A02D27F55F1DF2EB6090F94B64076FC2D3D3B3694DA5C7E"
03489     "B2A180DEA14AEC21156E1110DF75616685FD53C72252FB87E7D19EF5AE8D9E129D0984A065"
03490     "20C789DE22CBF6E2271801691E0C3CC672F6865A41559910D0279AB9E0112E66B2A1C2B22B"
03491     "6679CF70E3FF870084562ADB36F532E64D44C7264E44D488076F7714A1089526118D4655FF"
03492     "E16B02D803AC2601493CEC3AB27878CE95ED3F321913217DE12B8E5E8FB75A85707EAF6F1F"
03493     "F08CC86B91632ABC7CDC42D1C5D0F2F49D5F412B4AD0C5C46CFA74643C9333C5F3558DC3AB"
03494     "DBCBE23A3573146D648D540116136F3F29E42FF07E26C1504E47FCAD1FDAE63357E421E464"
03495     "40424921404ED02FD4BFA2FDE68CBE6CE49E4E120C141013DF5C0BDF776EE36FD5899C0FC0"
03496     "57DAE2C9DAEF6D37BF8E85258BB36B54ED8374BDDD49B6AC2C8BF3105A194F76DD512336EA"
03497     "AFC7BD2054AF6A9606517DE03AF445CCA5FA65307D2E116E42A3676EB6033AA17D76A87F52"
```

```
03498     "D144CE25E3A8DAFD3E044289C4600BFAA9CCD963D14A19C5911C3DE649439440D11A21154E"
03499     "F25649F7DA295E0FDAE8C48351BC005C011A10D201B3062492A7CC933AB5D854B26232B709"
03500     "1CD0B7AA3135F28E3AE75E267C223C5E03B60FAC1BF78123C5AF76719191CB6BA277A5BE81"
03501     "E64117AB344D92837B6D600F36702F4BB4532C5EB1C9BAD8A111C540F52A225DACFAB37641"
03502     "898B1FF770A523F8BC8BB0DADB59235DE055810396F1993539A9CAAB622389607DCDFDD51F"
03503     "B67CBE89F0F868DB4D27A7A38542A076D158919DCB621BD325F23AA0DF694C444206FD4219"
03504     "2FE7F9A05743CD54D8F111676AC35A3230E372A5A6D7E213C4584EEA1A4993067FD28DF669"
03505     "0BE9D3E94AC06BCF89BF1AA47496F8F6A18524187BE80D59A4E80193CFD757B706AD483A91"
03506     "6AFBE2A56E0A69F3B1BBF9F4B239D05C6C556A8D22B00E9BB1FDAA620D949ECDE86EAA299B"
03507     "E93A7884C99FA782F2BB3BEEE86046489B3B8A5930",  // PRN 48
03508     "71BB1B2E833793D854F8A9A81E6A6947057B9571F2BA99380DDB25D878D6B48F09ED7DBFAC"
03509     "E92B6B82F413E038128F6128AF3BC467E9A4DF2861DAAC674B6D948A10F28F7D43657FEE26"
03510     "577AF438A2F4422186930702EBC6C9173E661D59CE7594DF95B861F9D12EB060FCD3BA4315"
03511     "9C9A1BDC1EF13E04893E411267331588CF4831978469FF569C1A738C54001BB5CF4FABD289"
03512     "075A165EDE0A58F6CF6D215D306A7840CBECD0E87E3AD186F7A67A967373551E13D2956E5C"
03513     "578A7F5BD50E2D570F9B914848D46A640913EBED2E2ABFB86916BC34EEB3E8A671AD771F6D"
03514     "3780B6FEC143E26F53B02977255314BAE9A2CD9E5BA2B49C73226FDC724A859F8BAD3A9FAC"
03515     "A1B0E5F2DFE7E1E45DB6D4BE2B76535A817F94594A4541C01BB62AA83A690B6D84FC13B632"
03516     "972C61D940F2C9D32837413AF8E42045ECA3072CD044B2183400CE63C418879D13FD281A8D"
03517     "0835256DDD2BC3C9750C1D44CF1037FD7264B7716398FCF1D31CFFD0B7C52C6370E4CE6FC1"
03518     "63B40436490A757465B20B8890C3B5C0AFB971ABFD01796569E3BC73C13D1E4B1FBC1CDCE5"
03519     "9D21B6B110272E5770C589603FE67779A49AD0EC66910A2BF4D8C8ECC18A32EF92F502126A"
03520     "5DC3DE618233B9914A9608B2F17E51611150D9A3BEF1D5701A9D465A1437DE24371C9179800"
03521     "CB5728A7F3D734A2A706BC64D356BDF591389970B6CC139AC510A98E3C75F20120450CE453"
03522     "73AAEA6A279BBD17221BCD32ECF82C11B4C1CEE0A44792CF56978D3D2399F7ECDE9F8D9217"
03523     "F8BA22770E210D0F1AA852178B872E296762873765A73DEC08873C04ED69C995C5751B97DE"
03524     "C23B94CA674FE3F66211317B074D8203EC530A20B6E6DD21AB55895BEE1CBD0876183D652F"
03525     "4A0D2EFC95749F8F192F860BEA534598FD709B396C209CEAE4D9190980733E7E98C8ABE52A"
03526     "53C68D86053B56BA6FCAB5C827292D729CB8BFD1FC8CAFCDE15E4527B604018F28AA16C1E9"
03527     "13F55461AF87C9A7BE1A742002E52B3A14EC30B259DDE7BB892CEAF77D25B7670B339B3348"
03528     "78F697C00CE6740117AE7C67DF3F8A7BEAC89D4872682C47F368F835AFD7ECF0D8471AD014"
03529     "68B7BBB0A974EC469A8F79ECF8D379DC13685D2A8F6F19CE102C3DE34B11422AFB42E894C8"
03530     "D00F606296DABA7123FCE039ED27324D60E853BA94DC6384540882813350437A954333FF1A"
03531     "8D08E2A4D25CB3BA0D08BF6625E25EAD3C1EBFA2666AA49550578D3763ECDCE81303B53F18"
03532     "B00C8900AF4E0532C5ECBC94513DD9F50BE511CFE4D3DDBB3F112AE148DB062B2EADDB901C"
03533     "DA6AB6BF59D37F356AB34AF97D3DAAAA417642E87C9B95AA546C682ED641214605F82A4F48"
03534     "6C9C72576106F76D7152615EC8E77187D4485071CFC6B0AE44880442790696E057A3AE20C8"
03535     "60691353B3F6BEC5F1C2DA07563B423BC01E0334099571158A432441256D7C409B7B6EF264"
03536     "42075ED17E2BE37F8EBC049CFBE0FA89CDA7A58DD32C417B34E899FBE86E2FAB8D30846DA1"
03537     "7144A6A66AAE1C24FCFABDA5B573FD2D6337226B5E49BB031B4D2B455B6DB871076F67AC03"
03538     "C3A73CEC01BD0B1EC42ABD177127E62A66FE8E475B982B4490F0877466EEFC7317A703C5C0"
03539     "7937340ED4B53E5DE5325197FA31B8C8E05AA2222064EE5D7C06D4A1EB53151F75C94A2E25"
03540     "9688CA0716548465C5C255D81FF10BACC2C13098ED8CF7F5B15193EE14FB5D258E95EDCC93"
03541     "E9796FA823892C705A5771D561787C12592D269D657FBB71F021F365B7453D50C35F748FB2"
03542     "B7F36DF28769B81EF12A26A237FB0239C173559540",  // PRN 49
03543     "53DA0E7B84741AA9E225483630169ACBCC03EB8CDA28B7BDA685C756D66B14488A2D0AEF7E"
03544     "6CB2D80F2726327257B7284B93EA1B56AB80FAE668C04FA49FCD658D896A997685E1EDB4DD"
03545     "F85456B37F32FC8CDE50882EF0F09BE4ED4AB9A425806A49E8347A42A50B38FA8D1DA2FD2C"
03546     "9438618B6701DEE159060C186D50170F24F38E07B185E3272EAEA4A0A7CA41A69C69E9D95E"
03547     "271287D3AB8284146A58440EA131A7F47D73CB2BCF40FE3A58E1B998C2E5EF9CEEC8EA2F84"
03548     "67BB7757C03A99FC8F014EE933A7080CD46625A2A7A251B7A37E4208956A8C9BD35E6F8674"
03549     "BC06FCAA5DD04A2558C9665C7985014D3AD95ED256FAFA358962EF5BB26AE2FCE899392DF8"
03550     "58F99303E2417BCA7672E991FEB891F5DFCA2D461148367C5C0DE1460BF557194533DAF01A"
03551     "5E8E0E43D57B825AF7EEFF163DAA23B9F95C063A26B3D213459D885AA96023715CD21DFA2A"
03552     "2250F7610B78A77123443BD06EFB7DC85D1F16D0019D2937C3DEC4DE6389485ABB21642B6E"
03553     "41ADD43CE96F228C08DF6288A647EC2FE96032B6DCD651FF950B72964EE08FC2030272E3F6"
03554     "01DB7F7E770E655389CA6CFA2F9B87CE76FB0E0CDCA4EEE5E80FC756BE46CC09F84BDB34AD"
03555     "A2AFC024ABDE0066ED939F8EBC236CB3F577C1BFD741F9D101A038EC86AB0A85462BAFB2E4"
03556     "84D6722499A6310FA449D979030B2A21206D44225800BE2228FA00AE6D92C8DA652E1B003B"
03557     "D2734D30557B735CC2A591E090394DB791245C22B4D29E706476593B6F90C694C5B87BBB0F"
03558     "A2C479E292A768A9687A713336A21D1199186F852C41F586E9BBC64004D8BB6814BFE73983"
03559     "4C99923177AAF87B926D56A7AFC0879C027332A60951C84E9314380A5A78E1196D094F15D8"
03560     "56AA36742825D2B397156BCAD8ABE7291FB41DB4365AAE49CA82CA066D3B4366D3122ABBC0"
03561     "0F05559DAFEBA9F98361DDAEF068D60B18265E7184C4D6BC9C3619CFF5C758090FF6398CCC"
03562     "EB78176D2A8A2A4B9854C4ACF5CB614DC1CA0E15E7E85442241D48FD3D6E851A5D3947FA76"
03563     "9560928948FA26FA16EFBD2159994BD92B3D6B0C62818C91D4724413A7F40B2A2D67F4FC97"
03564     "B5DF6A7E3CEC03158E201D6643F402D3DD6995A42900D46C2881198CAD28A27489F5116ECE"
03565     "C3E38D999B2020E0C381DB3B8230811270D75950D9BB61548802DBBCB68ED8C7BCCB50D606"
03566     "BE400BECF873498621E66ABD2AA179B3E90E055C3719CE2FE047F815B95B065BA086B467AF"
03567     "4124E276F8CEAD000BCA5499D36217B250009A7B43E81CB3F8B1A3238EE436FE61F2F94279"
03568     "6DBCBE570BB4FC783B35C3CA31BDD432B33AD75B08107253E8F910EFE0D0B5453A8A055D88"
03569     "4892278688B3ECA612452B590AF38DBDD9A7070C5610E7A3CA6C91D24438E7F45E7A2A330F"
03570     "164AEDFFF1789D5E875EEF121298DB79C77278ABFEC3FE3DF843C46F40E847272EB2669BAB"
03571     "A808C38E31F13516D5066AF4DFCDE6EDB2FF0B0A4CE9FA9B4101F6F144B02384868617CD39"
03572     "175852E065473D6F566CD18D7403FFD24DD33ADDB52C7CC22167E49102C46DC369A92CE2D2"
03573     "FCB81B4D1F14B7CD2F80A65D8FBD20FDAA23219873ACB8CF934E68D6F8FED6B41193CFAB1F"
03574     "44CE4BFC7C67DE1E8804B47DFD7E8AE281E19846AEB6FF94AE7E7CF6FFAB46242843811E6C"
03575     "5BDB78157C76DF4F92FD3653D7FA5978316EB055059C6A2B6306C957418860A88F63355E76"
03576     "D96F4727128D9B3EB98501AF5B093F2C314F98EA2CDB89468E1BD51138CBF25E8B911C26B9"
03577     "7DCCA47F1A1D6C1CD415A5079A756B8A8715DD3164",  // PRN 50
03578 };
03579
03580 // Galileo E5a-I secondary code
03581 constexpr char GALILEO_E5A_I_SECONDARY_CODE[] = "10000100001011101001";
03582
03583 // Galileo E5a-Q secondary codes
03584 constexpr size_t GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH = 100;
```

```
03585 constexpr char GALILEO_E5A_Q_SECONDARY_CODE[GALILEO_E5A_NUMBER_OF_CODES][101] = {
03586
      "10000011111110110111101101001110110001111011011100001010101000001000111111011100011001001101100011100",
03587
      "01100110010101011000101111010011110011100000110001110111100100101110100000110011010100000101001001 01",
03588
      "01011001101000000010010110101001110000011010111100000110010100011011011101111001101010000011100000001",
03589
      "11010011101000110010011001000000011110000010111101111011000110001110010011011110111010101001011011 1",
03590
      "10111001000111111100101011010111011101100000110000100001100011111010010110010011010010001010100100 11",
03591
      "10111010110001110111111010010011001110100111011110010001010000001111000010010100111110111111110011000",
03592
      "01010011011101111000010111011110001010000000100100100111110001101011010110001011101001100111011101 10",
03593
      "11101111110010101011010010110110010111110011100001010011000111101100101000100010001001010111111100010",
03594
      "01111001111110001100101011101000001110000100011101011110101001010101100001001011111011111100100110 11",
03595
      "11001010010100010111000011111110101000111010100000010000111011000110000001101011011001100100100101 00",
03596
      "00011111110000110010010000010000011001010010101000101100010010011011110110000100010111100101011001 11",
03597
      "11111110000010101001101001111010111111011010110001000100111001000010110010111001010111010010011000 01",
03598
      "10110000001100000110001011011100001010110111000110011001010111010101101011011000101101111101101111 10",
03599
      "11110110110000111001100010011001001111110101100110001110001011011111010000100011010111010011110101 01",
03600
      "00011011101100101111101110001011010110111111001001000011100101011100001011101111001111000101101000 01",
03601
      "00101111100100100000011010000111110100100011100011001100011100000100011011101111011010101111110010 01",
03602
      "00110100000101100011100010000110111111000100111011010111111100101010100100101110111111011011101110 00",
03603
      "01100110101010000111001011001110010001111000001100111111101100101101111111010101011000100101101011 01",
03604
      "10011001110101011010011100000000101100010110010010010000010100100101110111001110111100001110010101 00",
03605
      "10000001110101110001101111010110111000000110100110100111101011001100101111101101110001100110110010 10",
03606
      "10100110010101000101001001000000011101001010100111100110011110000000011011011100111010011111011000 110",
03607
      "11000011001110010110101000010000000110111110110110101111011000100011110011111100010111011101100110 111",
03608
      "11000011110101001010101100100001000111011111001101101111001000010001000111110010000101000001110011 01",
03609
      "00111101111111111001001011110101011100111011000010111001110011001001001100101101011111000101000101 110000 1",
03610
      "10011001010010010000100111100000011101010111110101110000110011011110001110001001000100000010101101 01",
03611
      "10111001001110000101001101010101001000101101000100011001111101000000110000100101111111011010111011 00",
03612
      "11000111000110101011010100100111000000010010010001010100110111000000100110101100111001000010110111 1",
03613
      "00001100110110111000110010011110011110110101001111110101010111110101101100001010000001011001011110 11",
03614
      "01100001110001011111101000100101001011110001101011111000000100010100010001110110011001001001010011 11",
03615
      "01100010011000000010011101110111100011111101001111000110101110110100101110101010011110100101100111 01",
03616
      "11100111010001010100000100101111111101010011110111101011110100000011111100011100100110100110001100 11",
03617
      "00110101100100101010110000001000001111110011000101110101111110100111001001000110001110010000100110 00",
03618
      "01010010001010000100110110010100000111000011110111001010111100100111001000011101110110110001111111 01",
03619
      "01110011101100111101100011110000101011010101010111011111010011111110100000010100111011011000100100 00",
03620
      "10010100101111110001011011001000001110111101011101000110001011110110010010011000111000000010100000 10",
03621
      "10101000110000111101111000011010110001100110100000001000100110110000101101000101101100110101011110 01",
03622
      "00100010110101101110001010100111011010001110010111110011010111111111110010001110000000010111100101 10",
03623
      "00100101001100010000101000000110011001110101111010110010011100011111001010100000100111101010000011 01",
03624
      "10011111011110011001001111000110001000011101010010111111011001000000110100000010100110101011100000 11",
03625
      "11010110001010011001100111101010110011111000111001001100100001000001111100000001011010010100000010 111",
03626
      "11110110011001011010011111110101001000100000011011101010100100111010100000110100000001000001111000 1100",
03627
      "01000110111100111101001100000100001111111001001001100110111101010101111010110111101111001010101000 011",
03628
      "11100010111000111110100000100101010001100001011010111101100101101100111011111100101001100101000110 10",
```

```
03629
    "11100101010010000010001100011010100000101111100110100000000110100001100111011011010111100001101100 10",
03630
    "00100110010111000111111111001000010100001011011110100100111101111000101010101001110000011011001000",
03631
    "00110110010010100011101010011110101100001111000001001000000111011010000000011001100111010111111010 10",
03632
    "10011000000100001010011110101000100110001001011000010010011000111010000011110111010010011111010101 10",
03633 };
03634
03635
03636 /** \} */
03637 /** \} */
03638 #endif  // GNSS_SDR_GALILEO_E5A_H
```

13.698 Galileo_E5b.h File Reference

Defines system parameters for Galileo E5b signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstddef>
#include <cstdint>
```

Variables

- constexpr double GALILEO_E5B_FREQ_HZ = FREQ7

  Galileo E5b carrier frequency [Hz].
- constexpr double GALILEO_E5B_CODE_CHIP_RATE_CPS = 1.023e7

  Galileo E5b code rate [chips/s].
- constexpr double GALILEO_E5B_I_TIERED_CODE_PERIOD_S = 0.004

  Galileo E5b-I tiered code period [s].
- constexpr double GALILEO_E5B_Q_TIERED_CODE_PERIOD_S = 0.100

  Galileo E5b-Q tiered code period [s].
- constexpr double GALILEO_E5B_CODE_PERIOD_S = 0.001

  Galileo E5b primary code period [s].
- constexpr int32_t GALILEO_E5B_CODE_PERIOD_MS = 1

  Galileo E5b primary code period [ms].
- constexpr int32_t GALILEO_E5B_CODE_LENGTH_CHIPS = 10230

  Galileo E5b primary code length [chips].
- constexpr int32_t GALILEO_E5B_I_SECONDARY_CODE_LENGTH = 4

  Galileo E5b-I secondary code length [chips].
- constexpr int32_t GALILEO_E5B_Q_SECONDARY_CODE_LENGTH = 100

  Galileo E5b-Q secondary code length [chips].
- constexpr int32_t GALILEO_E5B_SYMBOL_RATE_BPS = 250

  Galileo E5b symbol rate [bits/second].
- constexpr int32_t GALILEO_E5B_NUMBER_OF_CODES = 50
- constexpr int32_t GALILEO_E5B_HISTORY_DEEP = 100
- constexpr uint32_t GALILEO_E5B_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr char GALILEO_E5B_I_SECONDARY_CODE [5] = "1110"
- constexpr size_t GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5B_I_PRIMARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH = 2558
- constexpr char GALILEO_E5B_Q_PRIMARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][2559]
- constexpr size_t GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH = 100
- constexpr char GALILEO_E5B_Q_SECONDARY_CODE [GALILEO_E5B_NUMBER_OF_CODES][101]

13.698.1 Detailed Description

Defines system parameters for Galileo E5b signal and NAV data.

Author

Piyush Gupta, 2020. piyush04111999@gmail.com

Note

Code added as part of GSoC 2020 program.

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_E5b.h.

13.699 Galileo_E5b.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_E5b.h
00003  * \brief  Defines system parameters for Galileo E5b signal and NAV data
00004  * \author Piyush Gupta, 2020. piyush04111999@gmail.com
00005  * \note Code added as part of GSoC 2020 program.
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GALILEO_E5B_H
00019 #define GNSS_SDR_GALILEO_E5B_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include "gnss_frequencies.h"
00023 #include <cstddef>
00024 #include <cstdint>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 // Carrier and code frequencies
00033 constexpr double GALILEO_E5B_FREQ_HZ = FREQ7;                //!< Galileo E5b carrier frequency [Hz]
00034 constexpr double GALILEO_E5B_CODE_CHIP_RATE_CPS = 1.023e7;    //!< Galileo E5b code rate [chips/s]
00035 constexpr double GALILEO_E5B_I_TIERED_CODE_PERIOD_S = 0.004;  //!< Galileo E5b-I tiered code period
      [s]
00036 constexpr double GALILEO_E5B_Q_TIERED_CODE_PERIOD_S = 0.100;  //!< Galileo E5b-Q tiered code period
      [s]
00037 constexpr double GALILEO_E5B_CODE_PERIOD_S = 0.001;           //!< Galileo E5b primary code period [s]
00038 constexpr int32_t GALILEO_E5B_CODE_PERIOD_MS = 1;             //!< Galileo E5b primary code period
      [ms]
00039 constexpr int32_t GALILEO_E5B_CODE_LENGTH_CHIPS = 10230;      //!< Galileo E5b primary code length
      [chips]
00040 constexpr int32_t GALILEO_E5B_I_SECONDARY_CODE_LENGTH = 4;    //!< Galileo E5b-I secondary code length
      [chips]
00041 constexpr int32_t GALILEO_E5B_Q_SECONDARY_CODE_LENGTH = 100;  //!< Galileo E5b-Q secondary code length
      [chips]
00042 constexpr int32_t GALILEO_E5B_SYMBOL_RATE_BPS = 250;          //!< Galileo E5b symbol rate
      [bits/second]
00043 constexpr int32_t GALILEO_E5B_NUMBER_OF_CODES = 50;
00044
00045
00046 // OBSERVABLE HISTORY DEEP FOR INTERPOLATION
00047 constexpr int32_t GALILEO_E5B_HISTORY_DEEP = 100;
00048
00049 // optimum parameters
00050 constexpr uint32_t GALILEO_E5B_OPT_ACQ_FS_SPS = 10000000;  //!< Sampling frequency that maximizes the
      acquisition SNR while using a non-multiple of chip rate
00051
00052 // Galileo E5b-I secondary code
00053 constexpr char GALILEO_E5B_I_SECONDARY_CODE[5] = "1110";
```

```
00054
00055 // Galileo E5b-I primary codes
00056 constexpr size_t GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH = 2558;
00057 constexpr char GALILEO_E5B_I_PRIMARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][2559] = {
00058     "C5BEA11BADD79BC7A91DC8DBED9A215B8FFFE4978A981C98CDEECE756A364B33FF851B7B3A"
00059     "FDEA0564CF9DA207C784E13542404E3BDE45C1DD70B4D6B51C88E3ED42A1C524E0BA5441A9"
00060     "C8766013E712313070BDFAD3699A226B719410EF18EDC843818877FEF748A0D7C6AA366163"
00061     "30A087CB91058D7F1A245311AC571864DF734F99E3E643AB7627B4CB57DACD23E4D22C5F2E"
00062     "3D2FBD4F04DF616E8110A001565B030C993141FE038670F574DCC4DF46C8DC18C84B449106"
00063     "9D1CAB4BB923AE6CD41BB9528ED40F7D139CD425416EB2D6342AB8808009A40C042F153DF2"
00064     "3C8B14F3F7235FAEBC9AFE0C845AD41F40E2FF7A7FB82971E014C9269C60AFE05F694AEFE3"
00065     "2B3194956253AC39AD42E0B7575F96CF2A094011C91A2FD3A10C02CB42D3425A3D6D85B404"
00066     "CDE1171ADCB1B5EE752F5783DFA14A7E26507C231FA5541FD06330ED44E37DDA40008A91BA"
00067     "74EFACBBB617C7EDFA3DDE592EA284381EF17477038A8B7ABF3C5BD673D658D11A22B97DA9"
00068     "F247FECF2143D69468CF8A112A69FD3E9488F14AB1506DB1240F26C52D2F19AE7A81A39E9B"
00069     "9304F65CB7FB3D9B4404679701216BDEF9F9E576EB36F7C9D77847A8FC40A7D7C4C65C8D78"
00070     "BD82FC7E5F90C6BE1CF19A2B7576DB1547A26C24E5F9BE97D2AED0A71BDEF74B26F92B0CAA"
00071     "A683DD3412BC52C7BD443F32EB3E39B2C9B1CD6B274C825AC7618CADE08D9991E5150281E2"
00072     "D9076EA58B2A99C54236A1E93F4D467692766080113C58FC53093C739FA8B0CB85DD0B6683"
00073     "CCDFC0907EE124E79C33C3EAC7CDF8E9F679B87B4FA71688BADC1DEE44FCE4952CAFEB0D3D"
00074     "6432F0775E3B6F34D18A24233A39C0B59A1F458B9F28B958539193A054DA2AC02B60D1146A"
00075     "1C6D5B86DE1BC424C2C7C703CAF92997E09A345AC0243EEB72468B161E5CE268DFE2A76B10"
00076     "F84DECB01F5F20419D9C326E7AB0BA22BD4225CDD990DE4A0C7E8C60E24A385DEC7F342E8E"
00077     "5BDA079ECA357DEE07A2E1690CBB8C28247B6E13AB795D6DE97FA074596DEE8B5846E8D904"
00078     "39B61BD71C228D10C7BC45886288F33654F787B4D336950EE6CB2989EF35D14200B9082C08"
00079     "5C6EFCAD497A12A053543DEE352E4AAEC83858967441F7023B90E263E8476D53E23FCA4805"
00080     "99FF7FA69C006271E1353CC546E04FDA1D6E2EE8DDA422F25216522B9657FD3D02CBA10247"
00081     "98C00CC2396154E2DEEEEF4EE6E25C8C980C2B78991C545C09AAD403A0D3D16FFB8C485C15"
00082     "D93A160AB2DF8296B811A25030688196C7208D66269EAC5C17CC4F00BB35CF85BC73DEDFE1"
00083     "CE6C6547D6C7873135A351E4C4335C257694D766CFE21440BE04432BE32583A166569FA7C5"
00084     "ECA0DF8D478ECD43F44D914A707902234E9A9E5C68FC472C8DE48C5E96071A1999108D6F98"
00085     "00F8D1141EE18F376C9A422A73C21535D055080845586C931FE8600366FBE3C80522D32EAC"
00086     "06E9005B545F59373C64AA152226F66E100B183690347C53698ADC5B8FEDA8AAB5D36823A1"
00087     "3F775B295EA8FD2EB09C23B80A3E96AC1580124E027004CB0A488FB2DDCD739E8BAF4B47B7"
00088     "539D2D7E3E79FB67255F5CEDBA801026A6DF3389A205DCDE967AE8F0BC766A53465E46AB20"
00089     "A67B379B11546D66E385B963CEC31C4B832C62F417122CE3D158DCB5D042A7290371BA414E"
00090     "9B59FE6379BDEE02FE476C9194F27E67B6953CDF9FE738A7B8256B2F8789D5679A5A172C46"
00091     "971176955E98BE1FAEABEF94D2B7E08ACDD9D9FA341A47D211123C6DFB12330142B8171DDF"
00092     "404225E20A90DC94DC00B1627B704BFBF6264A74C4",  // PRN 01
00093     "4F62483A58B2AA45A7F0B344DFA1C95568646AE79869337332CF0FB02D7A9A91633F160EC7"
00094     "D50EB74D2FBD3050B16E2E264EFC804070A7845AB5903A986183DF0F7146DE5BC923841446"
00095     "46BDB3D46E3A6E436FFB19E16C324B001C4CA6D4B223874D15032F743481B2E4D8B0D77B23"
00096     "BBEDCD2ADC458C23180C1632887694F12949B95584ADF4040F21E2F47F1F1F5E4E7FC3AD65"
00097     "21F1D8CAD7FEC9EB2BC1CF3C04AB70E54E55A31F654634A1D29005CDF05797D3E7FD5B2B94"
00098     "398BCEF55A6D2B0BA49C79066A5866FABFF8D59582FA407E795189B3AFC33B798F672FD41E"
00099     "6806D877ED31A4614627F6D46011FC6153A9F3143018BCEE9A0F98FF48194AE7C51539C175"
00100     "B66C5AFE7C9BCCF8D6B4926122EFEE9AEE07CECAAAF783BD40AD8376A95A0345A3EF7DC8F8"
00101     "8237E25BFD6F509AFF85DE7C7101CF51766E6B4419424A10F15BF4343BEB845E5F6279FF54"
00102     "4577F26DAAE7522580AC9EC833D3021AF903BAEA8CC53B5F0A630639789CDA3CA6FC72F96E"
00103     "F17EE975366610DBE3DA5063955CA0DB7FD3318E14AFAA514CBDC864F40261D54EC284F510"
00104     "6A1F385409CF82CDB91D535B470A883B0D82167435E2533D123A7B71FFC8914A370923B121"
00105     "4F5341C2D65BC868570185F2472E93E77FAE64BBB112A2E7B63F2117214B9BD0B2D9029AEA"
00106     "EA8F6E6A70F18D178B0B4D26AF7F5FDC8CF149B6744EF28F77B7D87DCC92D0E2CEF3977D38"
00107     "3E8920F71D89846CA5DE3F4FD76D51A0B5B69A6589774EC227CEC64B16217EB55B37EFF5CA"
00108     "78B5DFBAAD361DE5E606627EBA04F5FADBA5224A6C38F63289E9874A56180B5C57DC188A62"
00109     "E1BC314488F06E728399066FB9EBDEBFCB2F20466CB0C6EE1B2B8B158B769C40409838D95E"
00110     "C5EB52E169DDFDCF0946700852FA1F10436E2C093E257DC969C5D3195CC4DA6E376CB1A915"
00111     "A92519EE1EB2034E3C55AF3C24C247D74832D46D532F9797268C3B0DAABD705945F4EC5514"
00112     "19E94E84A13E75AC428B2559C2076333DED3A6E9CB604751E2FDF379B9A1BD6099274E90B5"
00113     "23A19010478B0958A5C53A2E9A18CEC53589AFCFB075387425FB3A24C2F66B40E55DC3F52F"
00114     "C4048AC77730953DE72BE2E662AB41EEAAC4A661A6E43C65AD368FC2AEBB7F658C2D8A46C9"
00115     "8234071009ABB81C7E33A75B86D89BB3484391D210582A9407637D82A9DEA9E2BECD0B4CF9"
00116     "CFDFA6E0B7D377E2EF3E91212D212B2F9515E6B541046EE2E5744887EDC1A0B149F848E7D4"
00117     "EF9CD5AAE0010132C6C5A8A5E1082CDDD46EF100C82AB553EBE5490184971B1F08D110156E"
00118     "0BCD083B81F94D47F19C0F41C815333C34384F16E9B34D659E6A65CB9B274725F9C535420A"
00119     "A73CF0B1497C96EF000D5227834F7C8124B114F9711A0AB19D7A563679092093D5B8DF349A"
00120     "F5AC30E87FC737DFFDACB3E95981837F7C48F97F3E34BF278616C5A14180D81B3BAA654206"
00121     "42B74606D764F6897BCB2A12EA2A07F0DAD3DD1DEC00E265C0D965EA6CFBBD5E3721F4CEAC"
00122     "C09B3B04E4E23873AC76E64B17C071CE9A6E336676835E854D1E0E2B2618D918E41808FF0D"
00123     "B2296CE151A2883AA314FBF2E9A300A75C841AFCB3EC84DB34F392DF4DD970F5ECBEF9E81D"
00124     "9DCBA9EF78AF41CDFBD9D77B6EDC92795175D805137996CAB4F91FC8935EFCCDD788AE1F5F"
00125     "C650D4840EF222A78401DDD313EA4B4943A89DCE637F1F81636D06C6E1CC1062162320AA07"
00126     "D01F679F57384820286F29CBE9881FBECC3E1AE4D7FAC3A143046E29D9D4A19B95BA2508C7"
00127     "030C0E846E358D52284A921ECA6F2C7890696F6008",  // PRN 02
00128     "FD5488290C9EF7CC7C4E4F2A6900C2F9CB24239B239645B27D990BA8878505E1987F8DF3A7"
00129     "1105499EED4410AED6FDF8434EAEF0C2C0FBC1C2508710CE37EBF740B273BDFE45F4B28625"
00130     "9021B2182D6185078F0C7A6580B1371F023F07236F1655C9BE30F35F2485C9EEDBF030068C"
00131     "660A9E754C0B6F949CF5556C133AD512C65DD3E4AF6A01AC9F4B111F107FAD356EF7C4E0E8"
00132     "814527E486226D9A6FD2D9D0F1B7B1A0F840F99C83F8483E3DBA2F1B576634BD5ECC49640A"
00133     "EC7338185D72D629A03F12CF53EB3DE8FA86112F5D0E666DE0DC3CC772693897690EA2986B"
00134     "2B9F02C213FD46ABA0A2DA24A2D90CE15BF1E6069347D5A1E78BC55D13E2BE2D173003FF68"
00135     "317BFFC96A9BF858A07A4D2EC5CAD1D4FB21CDE30D67B1DE383096987B93F8C709E3033772"
00136     "32EE2692324D8C585E35B3E9978B13B8579B0E759B4CA76B6E69CFDC350C1FD02F7088D0BF"
00137     "BA5633A8EDAB83871C1693BDCEFBE6A8D0E39A78DBB3D190196FBAF7033238792139561543"
00138     "8C75D59705810BA4CA4380367AD26B08B85A19D167B0ECB27EB8600D9F6FB96375684ABA1E"
00139     "FE9460DECA0690FEA707B2410781C571ECCF567569E07B78311E5AFD6A3EABBCFB6E68068E"
00140     "365517FDFE9E1A15FDAC6A000C324D82FA7D451334B873888B6DD164C89CFF3253BFC2DEBE"
```

```
00141     "4CC3AF78C66EC6DCA4BD529C3DDBC8B932120733133E754F235354144B3EB4E37E30038F90"
00142     "73F211C432431B192E9605A63B39DC6C31AC889E468DF3CAFF2A8D63683235B935784F6055"
00143     "02AE8E63794FF8E27C7A11502C99A834B0BBEA2AAD4300405F9C373203DC05D28CBFF5260C"
00144     "001DB0F963792F6D1D1BFE1345A4084A8A0EFEA3183A6C67E4852B90A426A257279EDFC751"
00145     "E4D3963816D5044B16248366BCCB874E9572DC9CF65FEC89F8D8A29F7E2328A0AAE3FE1930"
00146     "1C7A440115F881A3808B64B8E3C17EC0F225F1AB1CA4824D54330E504F562B9981511392F4"
00147     "715BA02578130DD776C8023249FE78FDD66D92569D9D5E652EBBD89A5C55C6939BF99A3441"
00148     "E6E3ACC944033FC5560A4512C3EA1F78890064C1663EC86BFCC9626C30CE67487A0928E95F"
00149     "A0957B774DA5F9A6730BCB7022E74D5DF4A6EBBE76D27317B972CEA4C0FE3FE1517E88361C"
00150     "32E253C025B4F16F55D532A75C3F15FD1AD532CFB9BFC20AA18B5D957FF969185B55405AC7"
00151     "BB185CD83E5AB99C31D475B264954130BFC6318C437E5B437109818AFEEC7C2B78DBD7076E"
00152     "DCA94F0EF2E5EEFBBDB65B22BE8DA21CE7B83A018E78A671A7669B1F8EC62B7AC522886AD8"
00153     "4377108E5E55FBC79A64E68CFCCCB535538F892F0278CCD295F266FD1C84B33DAA3B340128"
00154     "BCBFA8BE0BE4AEE9E88344E5861E14747AB0F9DA7EBBC32CCF2424CB8CFC50627532607D47"
00155     "DF5E878537307FDB54276B92C3C3B0366623F2777C7196E3B6D8794E5A7A70D3C1CA7C583A"
00156     "4935150B37A8A562BAAA8C5B5B30107B1D887499A4629B1A6621F5D79F9AC3764C42B763D6"
00157     "03AB28A9C921827149922D9230581BB8F05079E1EB37215A45498B20667A25E613CAAD346A"
00158     "248F0140175E9A0CCD06BBFC25B8CBDFEA11B126DE738240AA33AA0B248C5D4AD67305E07B"
00159     "C8BE864647261512 47DF7D35D08B8DD373FB6A26134846B4DB32DCBF3053E5001A50B625E1"
00160     "2A8E480043E053793675DD68D2DCED8AE3FA434DC701CB8B96BC666BAADF5E1E70CD94B308"
00161     "2C36619E689A9CBA9CCDE701664DBED3B069242EA13EFCFFE6084846C0960D03E59C7451E2"
00162     "23622864163749D6B7D938FB0524B057AC679D3BA4",  // PRN 03
00163     "86277BAE531B790D4F98842B2F52D1DB87A7F18529C775D7B06A93D731CDBFF92DA0214527"
00164     "7AFDA053D99D96A7B14A3E7AA5657175CB63C2F4D724E4D66415F60390683EA64AB62ABE23"
00165     "78AFA4DA2830EDF2D10D97468F10FE82506175DE9AE325381EBA4C16F9AF3B59B6B8C74E88"
00166     "CF81460ED6495CDD869E5E64517FFCD6F6936F0BB490A964652B08052C742F66631708B303"
00167     "402481AD37387A11A8897054B9D5C8DF5EB869A17099361646D96CB62DA504AEF3E18ADE08"
00168     "0997F3C17AEABC5FCA015848CE5F4237F2D194241D6C9A93170E990C15FF9EFAC8485E8A5D"
00169     "458D42CCDFE68DD3163711FECF460A7AC531DED2FA12F81EDD46395575B7CB02BE0F8AABB7"
00170     "8D7D5587D4CCE7F282F4FF0DC5FB01AA94EA0E94AE6AA813F5E88EA2C57F265B8DAE1146AC"
00171     "809A2629F37B2A9BFE676D6A33F5C7B2A24A6ED8D05FE91708877BC53AF036D62E4F2A0425"
00172     "0EF4FD62626E22EA6041AD77B83FD9F675EFBC17C2F17D086949EBAE2612565FC14154D1C7"
00173     "13E67F72CACA978BF64D726DD6262BCBD5A92FAFC5DAC738DBA18640027F27B91D87448094"
00174     "B23C3285C8F3C41F5B6694F9EF10F3802FE515E3234C66894665362EDEA755195949133D5D"
00175     "08DDA19154DA4453A02F51AFB275E202FDB549E2AA72AB4DA011E5FAFE35AC8BE25CD4CB4A"
00176     "CE313395CB912ABC8907185960C7227CC10FF9D471B19B6E0B27A74E0579E262E7CBD62516"
00177     "1C45527C88A2AF58B1F9DED566BF97CB2EFFAB8EC777A8E38A3B643B6D90BCC31A89E07AF3"
00178     "A84E31204474487C0813EA7C8781F45AA81BF5A49F9159FAAE985C882AB9A411F1FCE97DE8"
00179     "D4E4A4A772242647FA6CB1DFE0DE4EFBC889E69636CF51C6F579B3B782FDEA3ACEF89A0254"
00180     "68AAA1BE57920A5A1C2D2390834D4388CA49E363C25E0B4AE63B421B0D713EF2EEC1388BAA"
00181     "CB7876140DBBCF88E6EFC1E9253534E38E94EECC9FC2CD6840FD5E7EE81B0838218CAC5138"
00182     "13601F2B1599CB6E6C5A08A438620A2143CCA82A40B2DCDAE2E7A342DECE1A930BB8E99A26"
00183     "223543D85EBAB5162F32534A67D99BC30AF66A97DC975C93B4955B741AC0A1B3A033B765D5"
00184     "769C4BC0018D5AA4FADFE595AF6C79E7D253539E3604DEC60B5E5E8CF3BDC59B185FD35239"
00185     "927D2CAB7CCC319A20549A8CF7FA5851DA35876C95C65B8DF4EEC2166536831579A8E2AF3D"
00186     "5BF01F6F14A2EEE03B0A2497C17A36FAF23261FF04A82B8301FD5108E0B12D910C5EED9772"
00187     "4F9C7BABF42390A0D603E25EFC734AEE71B526BC4B01AAC7890C77D4D260E23A86BE22D787"
00188     "9B194AEEE94ED3EE89A5FBAA7A64438F541A6545CFD3C43ED1CF805C40289A6CAB8E879D7B"
00189     "007E43090A9CBC9D463991F1B14FA4E0EEC840DFCC8293267835ECF3DD145465FFFE397697"
00190     "FCD1D71521435BADD8388CE0FC07B458AC0236C2339C53CDBAE4466C7FB94F507504827789"
00191     "268726BB2C5D9DB3A3F700F0ED6D6A412C05EA6D3CC03C900479EA252D87832010CA8E6E3C"
00192     "2FE98141B1A6C7A4FC3471E308CBC24DE9E134F6EAE1323AC962CC2497616E6FFCD4279403"
00193     "CA6388DC1902C0B41BD7EA8110C44E60FB5C08261C42735E91B2E40122BCB2966D22723516"
00194     "0F288E2AF24278926177258EC33CF02BAE9628314A0EC62622452252B2CD49082B05491B00"
00195     "DC0F7FFA8268448491F02F292E13CA2629FAE7ACC2617300A776483F06738E4EEF4BFE7443"
00196     "35980CBAAF37A4B9044B50D7EF8C5014B9B022D69F021BBC09B223E3C9C06ABBB3BC7CA7F6"
00197     "25E3E69F498C877008F9A53C6A2D42D1115105B860",  // PRN 04
00198     "9E39D51E661E0C75E62D1C3739E6B454B4971B299A37168E4D2B3788859300A530A857A078"
00199     "7012210813B9221A5FB377C42F6B198F057BCCEA4D34A71DCE22B893373A3D342C10BE1EFB"
00200     "B764C9AC9F1F65AD0C13C5872C87462D71F7B013C77A4EACC79244656CF890A2BB738B586B"
00201     "AA846476F5ED8C0898491E96ECF2A563978EE78C084482F1E00128339A07A511FB2A7E532C"
00202     "426A147A6D9AFCFB9E9BE2DBAEF57E9874B93F43499E3C2B46599DEFA6847EAA820B98DCC6"
00203     "6D1922428604FBA74A8C810CF6C9EBCA4A7455A39A88DE2B70FEEF95CA3A68973E0E288923"
00204     "E29761E76E73FB18B97057CF7C98996E91C09299F9ECAE5D754CF2C4729676473 65AB7A4A7"
00205     "65C7E3331F42C0E0F9CA5C3BEC59E19056861F11C68A36309C99B341142B2B66503F22C767"
00206     "0BCD3CF3D9548C346ED08F4384D5DFF49C69A9A8F99EA9D1157CECAF140A9B6EDC8E279289"
00207     "8344F9B76979409C8FFEB6DA107D32DEA5FDE3393D2CCD5BBCC8ED8499F58A611167AB89F6"
00208     "1119CB827482A81212B8EB2DC2BA0B67C2B84996801052AD41AB95B0CF27CEDA284AE01861"
00209     "153C41951E09152766BE75C2CFCFF447B30D68146E38C7A38A923CEADFB997998970CC56AA"
00210     "710976754762A9B29E7F860961659AF938AFE5A85B8FC169BAAD1F2126193643C05D07A822"
00211     "E30EF801BCB836ED8DC2A36427F773114459592DDE2EBC95C1AD12E7C91139CCAB8F45C2A3"
00212     "6278E93D4F5DE7D999232C9B0A69535A714E85393A7E2C05E956390C0710477A80EAB367A3"
00213     "4E389C5FF74CFAC5A940D64C35BDD458BE1EB10C484B0A25CF8F341ED44496A578B266FA89"
00214     "110DE962F2290E5A49DB622629919D7E81092CFFD2C3C918D23CC80330830A904DE725E0F8"
00215     "5EDB93CD9F253AF2703646E921FA79C0B710AF0341E555AD2AB5B818B497DD6EEA7D737155"
00216     "03080745DE8FA72FC86C9C966F0C1869F69CCA5942787EAC83D26DE003A39959D0C5975A67"
00217     "51AF27A7EC5F14A40A2E1DDFDBD0FED0A186BFF687636416638DB91805B8371DE10B7A5983"
00218     "00E36168ACFE39D1A98A15DA450AC97B59A66AC0DC1D4DDB7D9D3851674E1C00B9E15A8E0B"
00219     "3709C40A30F9C5CE51A41907685BE086A69B39BF5CB085095FE242768664FD2341EFC24710"
00220     "014D6806DB1EC351CDD374285EDF813E62E6263549E7AB98CDCDD8778E2F8379494DDE052E"
00221     "053C53A4E6053382E9049EECD45945DBF4B6DF0CFA0769B538EF9B176463EE8A9FB6C8F0AC"
00222     "C9480DDEEA5AD146491F29FB5B243974B6073E8C15FC3672146069C1A363418C5E4549CC3F"
00223     "F37925940B1E9409291BE4956991FEDF1C6DDC18ED7966413A67A11794271FB56A37401EB3"
00224     "4E4F7556C8A2DD9A9E854A4A3D29B35569F465ADD226C3023F698D7B64E8E7C29C2DD57FFA"
00225     "AB06ACD7F3E3E0C2EF8A6DD13CC9F1B629CB4751D1592551152911150 56FBF615C7B738F93"
00226     "0A3C8ECD6703837EB33EDA12132B99C30388C5B8838671A315F312DC50997FE0A77FA7530D"
00227     "7F66B70153AB60D80FEBBEF5747FAD22CB93F743781D35CC85F4DE1819626C4829EEE0881A"
```

```
00228     "EE1FE4D2BC59E65113DD8F1103C762B763CF7FD211D1AFDB80A2BD9AB3EC93A139E2EE007A"
00229     "3ABDBA21F590245898D25AB87ADF63BF0B3EEF5B33F4E18F984D27A73B3D77B14D178C949D"
00230     "EDF1F1D8E01BDB332B490BFB43094DEC9D0F29EC5CEC1B65B8040B7BCD04A8C2A7AC065ACF"
00231     "F8FEA5BE160C56E7D8B7B1000EEFE0584C02E2870B86D32F2946A4615D8C73C8CF5A634A64"
00232     "0B65CBE8A2A1276AF6C3D6715C4CC088FCA08BC290",  // PRN 05
00233     "EA7EDE7F64B6B80F492DEAA6BE36509A1DB8225760CDAEF69D6E682C6B0679FE860C2C047A"
00234     "24D7CD35DE983BBED286F2400E498D6D1D1C929DC0C9F379B312B41E293FFC1DAAE0D8370E"
00235     "57D41A8847926E68EFBE114508B67A92A28BF80D3ECAF1B8FA23E5B3D19908AB6DE56A856A"
00236     "D876019A6621B933AD563BD522289A8C13A52509A875FC8E9BE87AD629CCF6272B89005A37"
00237     "7563347D0B9719475C5D13736983AFDD5BBD923297D34AAF4DB3F63EB95AEC43D56945C76A9"
00238     "B31A7765D92F74725BCBD81D76D103F57D91B4184B0E52CD9612E388954C07D97C28A1B1CC"
00239     "BBAD61BA8EE8A8796959388EAEA7F36BEB2822CF5C650963463BA8C8F56862C436C566CEEA"
00240     "3F96845658A0105647E1C35ED73BFE3F04000C264A2B607CA48A9BED86BB1C75200323A525"
00241     "4404E9813EF14223F0D7F66E52AFFF952A903297A8ABBC1447F01F89DA82D403E8F9A08FCC"
00242     "A510D09AE803EAEC71CE25866435A9A64468C2F2187AC3A307755BED43B97533C07416141F"
00243     "8E5D90B421A087313C7976FA1D99DCA7C15C4FEE4F753ECFD4158FFFD4736706F0BB6B129B"
00244     "54450FE398F6E317706350A0614181254A839767DF7F30B31EA84547F89A3CD9458D2683E9"
00245     "46D2D7A91A12739251C86BB0CFA6BF2AB70674201B72ECFD0025EB8D24C872FED7A5D02571"
00246     "B67EA7A02225EEA2B7F65EF2727B5D26480956EAC7479048B92F273ADDA3E92D152ECBEF51"
00247     "0804631247F3634CEA988B7D58D47002D00D0923AD04B2D561A8751F947DE34C7459918521"
00248     "8804079CD7EF7DCD2C4406372A7EB63B91832EF3D24BF9C440FEC45CEDED9D8C5AA305B3C4"
00249     "7A8CF0687E08A94FFAE9308ECAEBB86F3F51A34C1370EE8A585BE01463259B409D89FA011E"
00250     "6FC64C4AA9E059AEFD7F43A2193440AEFDC0323D2F7294F2CC9B85F4F5DA336EA25DCAA666"
00251     "E16370CD53D3624674F097BA3D15DA343767CD721F2CB4D0A7FC716C16A4FA243DD8D89C22"
00252     "3AABBFFC4A4B45254EC40AAD1D15648DB2FC63E6A70832FEBA9BF10C1C19CA20C59BD7EB91"
00253     "D0ACDEDD004259AB134AC1CF90E86FAD39D630114A218118D72FF790EA29E906289F8BF4F6"
00254     "477DE5D1A585EE7054CB3195BAAC5B6380611186FF5BBD6967982A37894A55FE7BAD5F0F1E"
00255     "48D7F254587DFC7C7809A8967FA0F54729E0A86BAB5B2DC3DAF8AE12207CE7D968838B5B81"
00256     "E90B7E94BD88A79660E4107F970D3E86F77C29399CD2EDC31AF16847CA35FA221F6C113E85"
00257     "1728F8056AE6D87C1E424371408CE13EFC3B620ECCE09E774FE88DDD8B97438F9B236549D7"
00258     "E2BE8DDC002706326F14172C7673FCE8797D3E424A8F84D63C79BE8686C68074B0503F096A"
00259     "F9966C8EC55841F2122B8C1FE6070E487132AAD79E7790C70E4582823DE647E9B269C9F90F"
00260     "BD27F823CFA04228355943E9A0801D4CA513E1DA51B22CE1533CB2977DE21AA5ABF436DFA7"
00261     "FD4ACA03602DAE22D85672AC8995DB1C8B2E1938E29E892C0833B5D95381080EFAD71CB9BB"
00262     "9CADB337091956181CBF92A8FDA413DEA5255648D4BE7585F94F216ED82D3A4C31441F176A"
00263     "155F7F2FDC9DEDAFC01F7631FDA0F16106CF2DA29361F39E1847AAB0E5BAFB897621DB63DF"
00264     "B66A8CB39646B8595491015 33E3F94DC8FD26C272CEA922E6F901275BF96446D660788EF97"
00265     "EEB116A54FC63C6074D94518F48DB4AFC39465ADBF030F0F092FCC8E6594C2314CC366E8A4"
00266     "D876A2F39BACE218F6E66DA47FABF34EA0F7FE464E30C675AAA01A1DC26359E219FD52E90A"
00267     "B254D9A73C74DF9C1BC4C98C7E4863E33599389980",  // PRN 06
00268     "F283216A0505319CD7685323FD568BB1D0AD9B852327649599 73B2DB55620E2E9BC31B189A"
00269     "013592E220DBB8345C1F5105373D9786E74D5BF99309B25D1DFF61800F2685815DBD1CBC01"
00270     "DD9DC2D2BD8417AC887027BB5B85DA397D966FAAB1D8A1061C1D6875B5925879BB3EDBE91D"
00271     "8DEA9F874F3708262AC41FB31E1D58B892B3BF9C6AEF7C8C23EC2A5C6F84B53E006575B7AF"
00272     "BBF45B23F99BB539945A93CDCD6122312DE09FE27692E9A5D06AC135EF0B24DA853A95B36B"
00273     "7E9AFFCC95B6F65736F54F8B29BD3C8058D89AD336C8CFC758F3EC083C851F0A737069B2AD"
00274     "3C8484EA141ECA1F36CC0665B83786B83381395654512C94D9CED2D944A0F494CAE3F0D636"
00275     "A0D7AF68AC96B3E5EF120951E3C31862293AB7BA9C16F0BD267D77082A507C858B8B3EB893"
00276     "83DFB15CFDB2415827C574516FAA0F6CD20F9F7D2AE801DEAEE54A9A25DF9509B659DCA102"
00277     "A8DEEC84AA72635E8FDFD68E5AF04F7B980BF24FB44322889780796C6183963F85BBF38924"
00278     "90C2F0FF4B1E3CEF2AEC732164244BAE4F5A3DC13B5B75434B03A44B310D0DBD3B6DDC9312"
00279     "1FBA802349933E54C239A05609C7CFF25D71A0FBCE0359CDB352772E60166B1E8E65D165EC"
00280     "FB7B9E4090960A2056D8068B2838415204E3D75B62617AE0ACB20BB296ACC26F33D5C8344F"
00281     "1FA89F40EB060DC2C2DC87C55A3B0E0AC8914F8C7B830BF71943B9DEF52B6BE1C9A44AE6EB"
00282     "AE0082B9265FCAECB2A6199B0F195BFB65C215BF0829F4EE3267A422D9ADA363D8161070EA"
00283     "B8922BE03E6089BC3F579ACA97AB7F1D02EBFACD87EC57B6440AAF1FC5160F967CF8BC87A6"
00284     "3DB3311C18639C07DF20EF1C3C0C5F61805ABECE74DD7888D4FFC650FA223F7C892B9ACA85"
00285     "88F6B326389468C542262EA2CEC5933494733E2FE01D18811B0C869AE2EA3503AD1A2296C1"
00286     "1D61F68964D5D258F1C0E7B7EC09F7B03DC51C98A32F6E96B8BFE256B2181FDFD632397F63"
00287     "2FD52C461FAE2BFFDF2A9AA44E02D47A8678057493DE24EE3E11A2847D84B3FEF26E867A65"
00288     "109EEAA51746FFB0D2D26618122B233E5BE21F06C496D21047EB056FA7D8C97DCE5D77A400"
00289     "81FABF2423E5C92A0F4C45907165CCD1CF51C56A2B2728665CF0D58999E55FE8FB68A6F2F1"
00290     "ED9C649B7531273EF5F18CFAA384755AB1D73E9AC6707A7701C6A3251DFC9671BA29D025F4"
00291     "72F36E581762F8C41FF6B9C0EF006A1B8BD2166DD9CF89851A92C8BB15A27CCD597622A51E"
00292     "A7188D78B35BE1748466A0CC2464E0497E562F898C4523700715 25C8603582D50D05F9B11A"
00293     "3FD6BFC3F8750A301051BE191F36617D66F67C98B221F942D1F1C03BCCDECCAA82D566398A"
00294     "B534DEC485840C32E22E0516D9205B83EFA59289E66A747057AD0304E6002B14F6BD9EEC45"
00295     "8EEB6319EE9DF6F9EB4536FD39D0B9FAA0F94819A9706190821EDF74FBB69D893EAC0D3D49"
00296     "0B936D6A53D0E56696A0050AF484B7C5DFA6A5B45998FE7374BCEAD95C1111EDC742AD33ED"
00297     "64C48F5A5BBE257BDA93B941F8412A04BEDA8E888FCAE00F4376F4E53E16C782F45BC3553A"
00298     "FCB9CBB35230EE9C37B4233D22CE7610730C4521ADBE4578A34B3DFEDB1CBDA875D56D3135"
00299     "95B83CB8D3544C06B7CF341E6BFEEA5AF5A50D1BC439647739A0AAA98ED67B7103A33B9A75"
00300     "E66662F78485BA9AF2E1F2D15D7F9C13E12E77F871ED909643DB41ECE09AF25B11291DAA87"
00301     "2146ECBE2D3C39CA3A6DE7AF67599C422279E2645A238799D5AD93917C15261C71FE9D6F37"
00302     "7595D14198587B145E49ECB9D04C07C9051FA4CF4C",  // PRN 07
00303     "4FB0C9E163CF95D9330D8B1047AD6FCB28ADE705EC3E8838152105181C8164BBA0580529FB"
00304     "5E7DBDCBEE0B67CA9A1D8AA19B6846484B126A267D7F9AA3FFEDE6CF3A8261F435A483015B"
00305     "75FB39A6860FB9B7CC69D146ECB0025DC7E79A9880D0BEA3AE8A396B201202512C9C24F3EC"
00306     "2780318CBB04DB9D2323592CACD7E93D5452359D920C51A9B8433A1AD42623641C52A1866A"
00307     "05B914B2BC7AA0E906F4E4B931AB759546F4B793F5C006F60A7BDEFC6D79496E03E403D737"
00308     "2EC2385FEDD2E227D7E5EF1126DEC4EC14BD6960C93B8FD924C2023634C22A2B16542B9EC4"
00309     "1529F4C9354F4E2BD47A602FED2BE9FC91B6F247A2291AF02B9F9E3C464A8046F0F9FD0DCA"
00310     "9C118CD35C706B498483A0714AA401A504F3EDA93BDB22450D81B2CF97D855384B823AAFA9"
00311     "65A3FC320CA3EAC08AF2E7D00A204C5499B7A01DCF83AB47E161CF59B45D35E5D9E47F08FA"
00312     "60738C17A106A1D4BFF74BCCB4A897C8D3A99D1436472FE24A9B5BA56300A200221F619C92"
00313     "D298F908BDABAB6325BB2B3BE69EF7FA959633CA6169B2DAD376F212C93CA1439BBFF3493D"
00314     "40148EA772E191D1B5F90DB31B9D910521D57FBEC93724AABF55786368378ECB95C2993645"
```

```
00315     "FAB2AD13DAC66647B36EBDD0BFFC6EBC8BCAB78AA9BD2D7067D0ED1C5A190803A7D1F1FEE4"
00316     "CA5519A54D0D7C70404DBBD488536485A33136D2BD2FD1479787FA009F3EA1235CB451BDD2"
00317     "6B3B4EA79D2FDD6D95ECF2365E5A2EE06C8B52349872D77B32CCD53E5A28BDE3986EBCB57A"
00318     "059013153761EE8E783F5B34E088835D440758D1CCB8A96A16EA6CDC373453A584C3F06844"
00319     "C685CA3F2362235452317AC4F6B73AB623F9A6179B94F2A2AC59E3D105091EA88C4E524915"
00320     "5654770CA613E0B11F7D74C709D61A090B22A53B316D05E02D04442E92A524B71584D37E6B"
00321     "1BFF1841FD15379D5A4D8AA2FBDAA0786634084333234D89A95C5A4252B87CD12BF98C681E"
00322     "79095526996 57DB90728F021C5B3EF0D59F5D3A0B2ABB086373E35FB935C9A1432ED932596"
00323     "FE8182B668EAE731AD081E473634EFCC212EBF0C8A86841BCF5819CD429203B0CB548FC2AC"
00324     "F54B25C2F1A9B3F98F75CDFB0C0D50CF232158FA44A4CBA828CE2E2F23B5CA8E80EDC4F27E"
00325     "8CDC0C879A84DC821F9A7EF37FF285C599BCEDF34E4C07958C90865DE349847CE454BDC9E2"
00326     "3F977F58FED34CD2801373FB7B8E4299E90F88CE41E73F47B1C2B46CDA3181281C2CF6432A"
00327     "CACFAB5AFAA83BCB9ABB002FDBC0A0DD132F08BFE7276BDFDF827E36E4F7F1E4BBAB62045A"
00328     "420A72F251FBA2432B0701C0836F3DADCF6864C1AC31F849D353945021F7231A09612D46F4"
00329     "2C6E9A63C247BCD01688E7D6DFB4555A344C6E6D4183B6D497C837F6E04C4407540044B57A"
00330     "FEBE2296316CA0E2D3DC0F196248EDCCAD4BF4C8F2ED45AE2EB8EF352619EE6F79D10CE1BA"
00331     "7A5D6F34762D9A82AE98ADE36EB0E4529E081A11450EBB4F70AEA0378148F38C50DB624631"
00332     "2C7477FEC63B8ADABD0FFF3D1B2ECF3784D0337956B098FE667D1A86D6CAA1C24156EE3881"
00333     "EA7EBEEA9BD13C0D5CA57B44008815D32F2F0D30C41719510B4B167E9E1419CCF1E4FF8FF2"
00334     "CB61B051A5A1A3267D08D106D1B15F0F485B44785FF5D4CD5D95BA1CD4FB9CDB5DAD46A099"
00335     "7A3A4E2F8C0F2E536AC3A402B406F9E522D925ACE301274A2A0BED02F062F8C780A42D5F0B"
00336     "A170F4CCB36930655C4B825634B72D2B2524DFEECB9A171A97F2EA6E2BB43D00F89B92AD5E"
00337     "141818B58AEB94573472D636ECECF4868B7A761CE4",  // PRN 08
00338     "F0AB64492124BD092152A5EF9A1D3C1EA9ABCDBA0BE2B6E833DE5D291E7ED04B1E103AB01F"
00339     "ABE4FDDF722BCBDD8B7BB3E451FDD3D241019AEB2596F99632497E3A9D8212E4E35E523966"
00340     "AD1A028659087E2B2414260B9811289130ED5E999E123E739755529C59C6D5B53F24C912D2"
00341     "7757975FAB7C2058398F13DD51BDCD27E8A8EE45BE96998BB9905972F919E8F06B7BBDD76E"
00342     "C00229F58C2BCD6C50541D16AE8C8E5FCC411E5F9D1BA8BA4EE839491344F22FC41B4B561F"
00343     "F43076333B16737866CB58EC2F5A228F79E05E0A49C0364347911F66D2B48754A56C992765"
00344     "4615C47B096F4C3DD13EC8F83778369D86DE64B95F1BAE796510195D83A3CD8370F4D51079"
00345     "2BC2751A13A3EE000449C995463CDF824188591B453B88C6123E5063A1A9674B83350A07C5"
00346     "7EC68590BB99A2BDD3BE8753B17EAB98174DA3A4296037435D2A35A67C2D44B40AD017AFD4"
00347     "E32EF35D982A2BACB7D3DE110E1B530BFC9A03FF8ED64E1303A050FA21CF9A8CDBD9BE56F4"
00348     "A68D82EB0F5F5A816FACF7FC37263B7BFF8C691117443C3060E531B7EBEA83F369FB7F8EBB"
00349     "F8521E85A1B32982E5F95AB3918CD9CC29B65AC3478A97B36DB893BD29A20101A4B6065CEE"
00350     "814E3FD4ED5F07337B049CBA0BA7A1E837D64DFF70859529D7C1F835C9294CAE0003033509"
00351     "4DBBBD0C68D413C893ECE208A9A371BC802B5739A42FBD7772BB5013C1F2EDE8AE138CDF00"
00352     "658135B2BE5CFC6B115201C8F11818D97E414DB04AB1CCE8DAE9B2E0DCFCCB968617FD53E5"
00353     "195BB6B260127449CA276B6A1D1436C028194DA0CF1B1FB2ED8177645F543CDF6693B78FE4"
00354     "789AF52CD6DF731C258827E65C15455D8BD8EDC394112B63159E60194B3828F74E70334ECD"
00355     "EA8FCC80C0045536235806D9AD863E4E57432D82CDDEA283FEE84D43C5725F75C72468115C"
00356     "A91E29813350D918933C19403EAC3CD7AB637139E5D37CA0D6ADA2587E03D4A760A792A50E"
00357     "B0D3B7D8DF96FF968A07259E83FEDC1CD30D2ADDAC577031821708CAC3D3AC54D0E0345965"
00358     "214E94CADC858813DFCBABD5CA331980B73C83828984A3CEF59F05D466EE6FBE6E0FA927A4"
00359     "FF2EB5D1F28564611C0239F69FD1D5DE3378C36B1485FCFCE8521B431620FE85873F983887"
00360     "29A33A3E79174827B75D148A85D52C7F91B9187ABE72BEF39E5509C0D2B70A5E8627227E58"
00361     "86B3AD35E454866675DD3365F6E627869B4CCC0536C1B30F563A9AE01757282CA517041DF6"
00362     "362E7CBB9918773E287B51BB5F0833FDE7A88A2BAB3E18A3399D93A4CA71DC935086C21EA4"
00363     "78793FC43079FBAEDA897B326302FF4BCFA5C5295137ECB653035CD6C12120CAAF6F2CFCD7"
00364     "A9DE1B37987AEB506DC0243EADD301CDA8F313101DBFDEF6381A40BC5CE19F5569D2459F18"
00365     "08A0C406BDBEE678111E1F882839E1FF0270323658DF7B3628FE6037C68D2D174AC5F611E2"
00366     "13CF0372F3CF6B5444F9BFFBF0EF0DECBA0766C91A84A597A045FE874D4C18C72035E2FF48"
00367     "1E823E18DCA610552FAE185004D90D226592E361E98F250ACF0FC3637F51C993FE5263A305"
00368     "7A7D0DEC690F11A83653E023C12EF263211A17658F78D7E274490FEBC35FF38313B1C860F3"
00369     "61F21FC4624B2B85D180110AA5AEC4FDB886ACBB1986A7BB5B9B15A7E35CBDC81931310C04"
00370     "7A14BE6225BFC647DD634D1D010444917AF27768309E3EF1870DAEFDFA23788868110BF223"
00371     "12C2573DDB326F7093F91FE73B6084682CBF19BE60BC0B1303A4A326303D851EF82AB38ABA"
00372     "FEA16B676DE05B5488E43D0BFCDC2FFA46E5DE866C",  // PRN 09
00373     "79833B3A3E625E3B24FACC84F7C58A25036582B49F69EE62F0174F1A006431515E9B77FAFE"
00374     "E6A4FC8370038EFEAF5A3F71A7C7251E264A2D445C90AFA91A8CAC47372FFD16A8428C5CE3"
00375     "7E3BAA03F3D9111676E455E24F75CA70562D1ACD385EFF6121CA19577670CA0FAD805E776C"
00376     "94191238DBFBE00CDD037A5AFDD9CC972B3C1CEDF3EDA8B8965AC6077251D219872F301BEA"
00377     "DA10D23A7696C5B4B41F3568BDC7823183E18BC0E700F9639A91DC4E893881015286169874"
00378     "EAF816F240E001AE137B20B9EB54AC541A0DF264A861B7FEAC185E794C367F2859234FD224"
00379     "67CB210F83C5C5C27F9E4DD7674621FA80397AB67E236AA01A18E0D99CC13EED9EC54D4EAF"
00380     "3732E0D26932AE9119384FB7A1CE111FCEC7E93EF064D5291974D28E1D77ACCF443633107F"
00381     "69E343DF308C78CF6C8BDAB65A256B5CB2B2644441D2D9E284074DDD486E63DF81290C61DC"
00382     "5D1A4C0B65331F60F1F83DF4FA60A009224EA70B4B9472CA00286667D200E5B0822B6FEB48"
00383     "2F468C3D44AF45585DF4BF0E0F2A33E4AB67756173D5CD805A705D2C7FCAAA6529B7733DE5"
00384     "9A1FC15597FC5B96A1B83CDDD4CC5D117AB42A16F75CC36341A7DACE024494FD7E518812BC"
00385     "17ED227D0AB7999593248B6C2C675B0DAF9D5BC104426F115E48364B17C3A83D49B7692A23"
00386     "9E6C840E12E1B25CEE25EF1B847FEB8DA3B4B783971A27D3605FBFDABA811E6FF196021E33"
00387     "B34A34F4EC4FDE2792801FC2A0287D148DD93206020BD45D05450460B704225BB2CE7B5A82"
00388     "AE9D8F59AB93C34252245DEDBEE02E93D6180E82D49847BDAF194F62B7DA2A1D63C2DF7ABA"
00389     "7977D6237716EE4E7BFADA13A965B4FD0167A41EBEE39C0E5D69281DE2B3F08CD4320007D7"
00390     "8184C4C2852D9106EDE4E9C2572EF6B4375A37BEC65CF18E1E241EA80413F93C763E7F9A9B"
00391     "F2B7ADC95129A6BE6BAC24B35FD214ACC820DC4326E36F0DFB6E185A9BE71F7E473FF8D364"
00392     "8E9930456BE9209EDD822EDEE10222EB01A8E1BA3AF11D90C4EB1AFCADC1EC5A1CFBFF21D3"
00393     "B4E54EA74383138D1FF37FBC9F1EA6BA7AC08BE4DCF18C67CA16510BDAE8C5F0D058B68D86"
00394     "719C96D1C3551A0ED7C14E5592C81813EB60F9D5F85B6078A5406A9DA454FE3E46093EFB27"
00395     "7B7DB99A07C342D2663DB9C1969B50197F0D0175425BB8F2A1F7DD5A6615C49409E2831324"
00396     "D18D6743206BD8C379FB5C7CB44ABFF6BD1D952016AF95EF2E385E6488C482C96A5C9832CA"
00397     "38DC27176D58B453DB4D2F193523B958FE4EC7B6BBA0C2CBD693A92E86D98EC8527396FF58"
00398     "195F872CA14F0A6DF0CAED78CF1F0E3CB4C9169DC8C9491474786705FFA996A6052B72BA8A"
00399     "2AE5B489DFD56B57A5CCD5A0670BC4E6D4C78D26B4EC6747DB0C5548DE1303B8642DA9FF35"
00400     "B2EA015BA154238B0CE0F8EAA7D7C0D7768368781F89FAAFD32BCAB1A93A6EAFF02442507D"
00401     "7C14702F7584CC4B17DAAF7A0AB42499ABEE3ECCB80DDFB851EAE421E7ACA7841D0E52613C"
```

```
00402     "777912B3C4FA7E4D8FB19662D680B7D361F959488C2EE6778C139887ABA8DDCDEF0F6A89E2"
00403     "0D32C1491349A0EAB14BD50D34135372DD8230435430BA0B120CCAEE4F26C40552F6C671F9"
00404     "6852E9FBE48FEDDAEABFEE3EA44907E35B6628C84B90F5FF795D873D345E8FDD01727F2762"
00405     "1EB1EFA4D1F5E53B6C7C2A9E674B6185B4A259CF7DBD9F515AB26D319818C786059BCD4B39"
00406     "80C8C14C4740B3E532C3633A14E23BA470C4C13CFB600B8342D70D41F4EB6A0FE2933959C3"
00407     "C6B733C35E64E0D77EEA6BB15548D7B21ADCC6AC60",  // PRN 10
00408     "EC2D915F0070493562F04F3549330482E0473736D3443D68D8D7ED6F0BDD523393B5FE4790"
00409     "EEB4C928046F05DAA556EF8F39556D48FF434563834A79F72AB1F1275D2517E7FB29BC64CE"
00410     "16F0B465920C48719622E618CC3D7518EB53A2F728B335DF34151BE0E239A3747072CD0A78"
00411     "88C8E7A4EBE7DD38FD84ED0E9344EDD98F3B3820BEC4E90909CC1CEA1C0EC2A63D139598AD"
00412     "F5618C795B1D5F0F86C81CDAEDD7651B2CB67DDB7198D4A5509B98BECE7120AB6F28977D0F"
00413     "F04B7C307BF10ED7BB2C0BB3BF14C138FD191D54D28C77A0A88575D7F97372FC35066ACC0A"
00414     "26A7AF5B816801B0718A37D666A492ECC2CD7255AA076F3CD30543DE651DB11C9FDEFFC6DE"
00415     "1A611BA322B1A19ED1B13F78412322A0452B32D27C88D3D4574094D97E5D41581AB8BF7318"
00416     "D4C56B76D214EEC890275DE64FB24952F1C8EF54EBA8C7C3557220B4DF722DFB77544A0D8F"
00417     "18DA398FBD1015B130C47B26181B67AB8A54DCB069F3715D168653D3865FA8EFA344ABB826"
00418     "6F7600410321A50BDF753335F804E7B95E3C0484F1CEE0D94110548669E46E88F6C94466B5"
00419     "1FC990F5B34206691AD89B76D6C0086EB50C57B6BAA273014458B32F7ED4EBCEB7BABA9295"
00420     "2FAC7A53287BEB1E060FA0583840BA971E4F9DAA9F0B5EFA1048D06ABA360A4D3206AB5C62"
00421     "20630B37C21A5CC60767BA826A6A638F9E31B623E4A4F03D93F610129DF0C5AD8D37AD5395"
00422     "93BBC37ED9174BC4685457F7D9DDCAA4D4ECA24201AD128D5F3A934859561AF9BB450D200C"
00423     "CE3F545306D2301D6C6BAA78AB8B7D0B07AF521A5DBF88D7536239E13824BB1F32773BE663"
00424     "D4FEF49C154B81B468F316D573DFDA280090D72A7845A6AF918472E284C295FF96947866B3"
00425     "F9C4C0EE8FA4C0A95943570079FFDB49CD3F5EAC5B43FF4D95F685230409A9071151D86909"
00426     "6D10A75A50BD7576729D98FDD5257DC835E18945199CAA55978C7CBD0A897E2E9F09322021"
00427     "E0EDA0BFD971E199AB3FE147FCF62598CC6FD9B34266E86F540BF85F0BA93D1441FD45DF47"
00428     "C6BC5A5753CA38366A94F63AC1799B6D0BE248064E879AD94F88201229657A641CB09C7A41"
00429     "14757515577AA38174B0D75746FCADA5073BE4009F7CE732E0EC2FFC40DDE7B14B79B3DB62"
00430     "DDD8C93CA7C3ECD3721F2853865D1AE262766746E7806F4E22DCC2CE04FCFA415F157A36AA"
00431     "BEBFA5AAB0BF9A16114CC2D5CC734B1D0F6CE593768DBFBC0858D73FC761675EB153457B98"
00432     "736EF866CA1A07EB98AD056B73177A8ECC4E1FD771D996ED6B0B30B96396340DF66EE20972"
00433     "5FB9C521A832ED90AE4E810AEA8B3135114F21BA188386A979D08BEFC35B156B3B12E5F1C4"
00434     "84028106BBE50BEA5EF5BE9ABF37EE04D380E097423A50370BCC8BBD1AB38C772551978112"
00435     "5990A1CB4E2699CE59A4073DF795577F6988DBFB6C06FD3F4F49A60E2D46ACA0415DF52765"
00436     "B03B1B210DCE28765F7AE7F4B27735EA5289700B1FD5CD62FE4413A442EDD15A726B9002BD"
00437     "EBF2CFB856195988055CAF3D4C2E4526CC193B1A73CD1643D564C86B3D14E057CE9CDF4C1E"
00438     "80D5C51609160B943DD1B32E21E2A188E07BB12C3E62C069B312A66F8B3D8AFCB549585D00"
00439     "008C37BF010886FF12B002A7886D7EFF5D6267623E1ECDCD31A55F207EE8C2171166B40744"
00440     "18E84095F71A0D204BC520BC47F856A714E03766FC85F58931980527 97BC7E13F6D133264A"
00441     "47EFE80716CE6716C68C304E04E9D0B790B9B3F0DEEB14B9212A1105300C556097E170FD19"
00442     "F3B8A6AAB5F861EA643AA18FB1CE5E24F491DE3DAC",  // PRN 11
00443     "409B1134F8998401BB764652FEF53D692BC6803D4B470BEF7B23F53CF5DC11118836A648D2"
00444     "768CCFC08A79C7DFF43E1AD138B84E475E8AD831DD388402DFC10087D79A5E3AD1DB0B0986"
00445     "E1B8B0CC19D433EBD411AF02A7367D5AAE7A66C7E60FDAC4CEBDD31A8220B9487BF09E0798"
00446     "449B0C658A419489E79366C9C8ED6991ED4247864654D4FA68B234917F4D6FDCFE23863580"
00447     "36DB8E9CBFD686281EA26AB0D39DE28698C9A2D3261DDFE5326766491CD4EBCCF98EFADC49"
00448     "0C594A5E69B1001BA0E773052ABF1B55621F86CA12B4A3C9FDA8CAED348F7A9A6073456535"
00449     "AFF370E787C24F0E26959DF4E816B1ECF31D0C3A61C4189FDE1C8E13BD058BA27300634293"
00450     "0810C51156B11A5FE717FCD813FDA1043BFF3827ADE87E9F440EE8BF90E95857E691B97227"
00451     "7611F1C270DA24475586309A188C832435F7B1F1E78EA8DA17DEB8C4F92374DE573A6CEDF7"
00452     "1A1EBD102EBAF07E7B58541A16E93C077E141FDD9AC50FFF7CADD9769FB9E572B3DA71D0C9"
00453     "614C8A0DAB73FE092A21D2C999225F51CF0AF547DB8D7693ED0DA5F11288BD3C6F37E1C492"
00454     "64F243CB3BF46AC35E86DD2B55FBA7D0F2A3D7B172AE829F8E80760402E075F81EE9032376"
00455     "39B98ED1DAE70613F9E3C075820A7FCA02A5585982F5B9989FA6F142CEC5500375502AC599"
00456     "F5C98D587559E47CE6D3FB1D05B110D2187A113CB787E0BC69AD39678C199DA82FBED77C64"
00457     "3CA165D439A808F953E4C983B026E60FCCB0CC5AA1B29CBF8D6329B95D3DA0D0DEE4CE5D4F"
00458     "D264B087FFC66E0C3162809FDCC4B1AE7DE9E358DAA7BDF9A65F98F0C4BC9C39E93F560F05"
00459     "913BF1106C7C07F52FFD07DF7A7F2C1786541175437A599B9063B1FC6722118CC4822A2291"
00460     "3F565A398D96D5B11A0D7C661D5A8A8F39757DD0EA5C98CEF3B9A037CA5987A05C7279C9D4"
00461     "D2D36B386B027B1BF85B21E7472FEBB9A99355D0B8A6D688B80EC37156F2A6AC04D532B060"
00462     "9343C77B0F9EF08312B5323EC4E07D3CFFEB6230B468BED5FD9F00175790253E4F3BBE057D"
00463     "5930D0815AFA8F7A4237F4B315577EE080D6F220BB3FBA999925F1A005F552575F4AE63360"
00464     "4F1150B5C805C8D80C702222C754860EC27648C07FC9441E9975A8A9254264AB8493BEF99F"
00465     "7F2D33DC4F805BF98A4854595580C3F478D01AD0B13D01C09F4AC02BEF02C785F0244C0422F"
00466     "842FB939858B3F10D73299BE79CA375FF18417057B91027971546111AD8DAC02179A053804"
00467     "D9D3A7BEA440675C83852C78B20A5C0864B8A5D0E435FC20C003DCFD5E7095515A47B308C7"
00468     "EC25969D69D95A91D65CF5A4505E250240FFB52E603A831B40808158D2932D3AD316E27B08"
00469     "DD09512534B59BFC2DD1C816A0D09C3B17848E5D63FEE578E609A5B1268CAE52E66E1435DC"
00470     "A7BF12A4FC1529D5AC9CD624AA19FCC934F2E1CAE19809A7EDEE2C6C75595E125C1DA37BED"
00471     "8936F10D4F67C00ED83F3241142B44D2C1508512AE98DA632857732869AAD5AB692019EDA3"
00472     "6152A656BA90C586590617EB9D7F3813B09C840B3E741681E495D650BC5AEA51FE7303F74F"
00473     "F700A8D19F1E672159BC330889BA1A9B570649F15320D730F5923496FCC3677E2BE5506C55"
00474     "FFB2D649803F7C3E9AA7FF000D9F3F039245CBA83EBA2CC8511DD411B4C696BBBDB6E498C2"
00475     "722F0B8C597729E7E6FD2324C14F802ED50EF06F25830DB70D7F46C82DD7DB1AA0B68B706A"
00476     "4F19FC0197009C497D4296F1247417D8994B354DE873977EFF02C4676783BE31B734972BC7"
00477     "32DC70EBA5F6FAF125535FD3127716C67CB7F3E444",  // PRN 12
00478     "397E16AE84EAAA97262B8839E03D5DA85030F2EB3EF773BABF2D7BE79631BD7F699727F539"
00479     "FFCDD05E0881BB23BBE8EF901966AB33D42051CE7398012D8FFF17E0D17D4035F0E3CFD0F9"
00480     "E94134D37B275D05E589F1D4BBC6E892D4EB2656FB7E5CE6294F1D870D6F096DCEAFC3E893"
00481     "0E19E1D9548A104EE4AA7173DFC8AFDE66579DF4F961891F83E9F455DAB2A83C9691689459"
00482     "1BA3F5A8FCF1EDA3E7B9A4B86DA95DD065D0961F60363C00C61C954BC2004F383CCCC6054A"
00483     "846FD7ADA6CE3C425236CEB9A374158329EF991320ECEB3BC5A4C272606466EA04E2B4D50F"
00484     "3A9FEB73A7EBD94E6A992EAA8A5B6E3CC5527E2E25857EF19B5C0ACEEE07FB77C3D63792BF"
00485     "1759B63AB024E6850099B28A855EBCADDC17A68145C8694963CE125A08A24E699256434AAE"
00486     "FFDF8F1C9A3EB70EE1EC2FB0CB5BBA264FC210659A47A5C93A0222F5DABC9464640FF649E5"
00487     "FEF1746B5D497721FED69529C0223D412C073D91191711F52D9DBCBB6C219AC4FB60696516"
00488     "CDFA0A66539A333DF3A7DBAE2EE466BA048B424C438E40144EF9BDA2EC89FE1E74B4B9CBD0"
```

```
00489     "FCCA4523B5E2A8C60FEE2875541684DF536A4AE32CDC7F6A7BF087A4885F3C25341BB44E3E"
00490     "CEF18A78513F7D89BBD0896FE9017788478C9C694C495683650E39B8AEDC4A3F7ED279C879"
00491     "399A669914FFADCDFCC470DA5E18949A7A036E9A4F17DD4F08D3111CEF5A30D2EC8EC4D149"
00492     "FF1B6731DD63D1DECB8EBFE98E6C619278CF1352F63797FFEF9FD74E8F986BB74842A3FC09"
00493     "2E7686E9ED8660C02AA9F829496F6BC3E2EF78C61A9F6990F37227CA0940945FBC4A79A63A"
00494     "186F2EAC6AD6494B78E2ADFDF4CD0F17B7846F1A88B92E389D7A7C31BF3A2B32285CB6A6DC"
00495     "2D27848BF9966E702CABBFC64F84092B47A177253B3C3585E0F7DC5124ED9EAFA05B7FC8EB"
00496     "7007F18CC9CCB1943DFA4C9B101121CF6DAC0B75B480B991FAA25B0170A3FF8924BB145018"
00497     "918743E49C34154C59291D02CA1226900BABA15D475EC07797B48AB24E7668A35FA5646D92"
00498     "570A5ACDF2A8D478B763154F7471C60811E003E3917C2CD3353800D77E9981E3C6B4439147"
00499     "342A838B40B3A472482E647F446F29B085D9C8C7B7C5B58053AD6D825976FA9D2DC007487C"
00500     "6938C75EBD1CB6F475A43474E246FA1A91EB68F80E2EFB6B7B42239684DF2211451241DBD4"
00501     "51853F5632C887AA3686D8211611440277CFB01A28B212F88B0F4864BC64F407B5137F17F5"
00502     "76C9011444FF2461B835B20CDBF170A37CE4CBC8442A7212125A660C5A1B2F783FE6707584"
00503     "F07E724990CD04808B55DF432711E9A73AB9046CE722B635B5BD20492E0B0A1C085E8F926E"
00504     "98CC54A94D821DBD6ADFD91CA9706A049140480258C11A4CE7EE66AFC56C2A21B4784671FE"
00505     "612D8873FE273CCDEFD2FD42CEBCAD0FC0B8C2B650876E248BA10978BB0970B5113E02DB30"
00506     "36F53D6F74D8CE6352F98B5B8621D2A35D2259870FA2A6BE07D5CCE435D10D29F2FC197DE2"
00507     "12FCC1926C7FD49CE08CC492A56960B783183F88C87A403B4D012E18E063F27BF0B5F82D75"
00508     "688C2207962ED06D711F31815D94FF16DC32F3D7A698F770233FE524D0534F4D681F2A2574"
00509     "A4D6F3E91F401767E2670A758C3714A857086768DE0F8FE428845344D15915A1725CE9BA3F"
00510     "D0DAF16CB1349ECD1EAA5F2E1F1EA58B3A793A22D1D37EF0DB0F86B8D9075904FDE73104EF"
00511     "7589E092A234394FBB3CCD9A91CD6B9A67A3C7DBE56F2ABB860E72490D6F756D11FDD7685B"
00512     "98612F33CBAC9A463E7B76C0D36A3040D44149E3D0",  // PRN 13
00513     "1E0FCD6CF84B05DA800B79CE6A08053DB3CA26A30B8E860D0F744E77D2321BB9B9CD586211"
00514     "119FE8FC010ACD5935884D4B13CBEAC08B1E1E39E68D6A7E172EA0BC49F467FA9A1FB4A67A"
00515     "66BFA5D12A92B9993C2478F604A4B5FD2EF82F9A3DA4E688FC4EEAF439A9AF576F632EB26C"
00516     "686884AA162C682F80D23177F3C2268FC9E33A4A3B4F58327E13913CC82FB732551ABA0FD2"
00517     "1BB9A5939B9E4A50752603B2FEA8C903ADC9A7A32EBCCB69A2E1541EC43F30FD64A02461AD"
00518     "1501B6FD7EF8F6B11758BA2959C68D422E9924DCB9F11F3CB669054A691CF11FFB725437AC"
00519     "9CF73A3DFAED63B0FEEB77BE7A4B27177E2719A7A3DB19CC4C2D280D0C958D617D53A9E3A6"
00520     "E4D409AE7CCCCC72C7FC5B6941CA931453B4D718471C5DFCD5224947DDC569EED813EF0FCF"
00521     "1B569693974E3A2EA44A8AC06DFBBA884ED632F408576383C11F5993E9194FB3D88CDD5671"
00522     "9C184ABE0BBE8E0470B9A7EB29B2E903F96F8A0FB06F2801C62FCDCBDF41B8EEA84609EC4D"
00523     "825FEFF18895EE74A1EC69715F432409BCC8A26631C874D798CBBD7A8D43874A5209215247"
00524     "1985FFE611FFDC3025A8CE37591FF938AE34CE8F5EF9EB5572550193AF8F76EA89161CA2D4"
00525     "F20309B56F06EAC2F57E6E5A666547E8FA77A50276916AA987097ED51F2A9C080E02B9168C"
00526     "FD3AE956558393B3DB2F0A7ACEF7E6203F562A6598479FD8A0CC1AAF6C29BFD57C12A6BDF6"
00527     "7B67A53E6C1A66A9D342908280142B7645587B31A788F54EE9C66B000483FB3A592ED0D0B8C"
00528     "8B37F03782E3A2A3AB3573F442FD88E4276595FF4505C76206DFC7A85BBF77AF88130EF6D6"
00529     "8ED313364B45FC8E846F4AF9ABD750BCC908F57CABACDFCA674785B0D05079484AFDA8CA5D"
00530     "E93CCA721E29F05CD5C361180B618E40625978C9A21011FCFCF1D81E8E981097921387 10F6"
00531     "3AB1BC46D9D87AB41BFC15342137CE27768BB27F96A00AA6D802196FD922C7E3FCD03E380E"
00532     "9E23FA716F7EEB212AEE3A2B8143A844F85FEE0E334ECB9C8454F7C08253E3BC65C69971D6"
00533     "93ED65A287DA9DDC6D561ADA1D9A5A892CFAD329AF98AA5AA3D64DCFEF5671BEDF2E8EC34E"
00534     "465D6736F94A67DD8AE904E6AF0AA913ACD059E1F62CBA857B355F64A0BAC546099D0B1293"
00535     "BF63876B4168506E196730C739A5E6EF742BB37DC22F6028AAB82FBF7BDF838048D3EDDC75"
00536     "1D993C2275C3E6A89E32A28E16E7D585A72B7EB3078B6C0271B730B4F93B50E9206A24A151"
00537     "0D6D0340811319AC992C1681EE74F4825E7029D02374059FEF0E02716A17A2663A2759F374"
00538     "965787A9201E2058065E1AA08AC52B830B85BBBF9B6AA8CC050234749FAADCA169C68644F8"
00539     "791155DBBFE7F12EA6EAE229B65529CA4B7FA78673039FECD69019D42C3ECF927F2B70A15D"
00540     "1F249344F7073BABE23741198469403 90F3B8F79309B041021 37D6E5DDF774863B84950FAF"
00541     "F25F52B1778DE93C79765E4AC178C32345CA66B0EA431D6F81592A02D9DD10768493573FC4"
00542     "9BC42ECB567F897C7BDFA5428C062165BC723AF220CA169EC4A6102B9739FEFC0EA17B2DEA"
00543     "B9E66F5DC749B52A93CF873F760B9A2DBE2D5C85208BC0FC8269A91CC37BE3DFF828395548"
00544     "A5B4B1A6D695658047A91DEB8B4E397E2D2806DE2794B0B51D9196165DAA336FFA13848E48"
00545     "57C7B44AE51DB34C64002FD02F8DF9FF72E242C344F2B5D58D017E0364B130DEB11863D5FC"
00546     "C814090DE66F0F1A9913EEB45050BC45447407D801695274E36210DCB3753A7645540BB0AA"
00547     "136BD572B2E2ECC0C18D46D60E4F52EE5A342B1A5C",  // PRN 14
00548     "CB2F5A4B8600DE451C3185E21C019B8BEBC01433929CE2139F841CFE511C74257A30EA27C1"
00549     "D4311599B509929E7BE1EB872380D77EBAD14065EB3536BE0E9A9DA3E936F80A9006A47A60"
00550     "A6C5B110EAA5FDE98A34527D430E45F247E05294714E747B6BA1455FBE24AB02952DC7E5F9"
00551     "D302C31056092471430EE366698AF5D791A12863756EDE47996B4EE98E6864329DCB5A53C0"
00552     "83D6A3F104C0415D476B8B6242F6F3DEE1EBE4B0E541FCAD230ECF15357CEBE779CBF2782D"
00553     "8868D8E0AEEFD997499186EA0A75DCB07AA5C97F47095D809E33065D8EEB4F8AD52968829F"
00554     "F1FD0DDD6B5E1322FE9D1760218E93BA20F4755F9707EFCD8A8E62A4616C5B0E9D109B480F"
00555     "999DF5E4DA312C479589A9D1BC9B8761C2ACD696C4EF23A2CE9F512F13AB072E19C618693B"
00556     "FF74FE0019BBA98EA19483755BE6B81EBACB1102A800CD08C47501099AFA78C67844A2B739"
00557     "AC11B7A8AF6B2B1206EDD9C622D90BBA0F3C3C2E4A9A0060F55707DEBB8239DC228F20EF3C"
00558     "B55C2FE7ACB112E9F6C2132DBE3AC47E2E15C5C437D31B4E2B3EAE2BD116AF856065C9D6B1"
00559     "CAC28797C5ADB972B770D99DFE101A319BF38A06E977F2076EE754F34004CD31EF8BB86D21"
00560     "DF12326CB19F1E87461B3C789457ABF1270A573F0A9154E9E0858883D903C7E9AAF30C5C11"
00561     "30B5F1E6FA2784273C476585ADF68BF2697CC282CE68073757F70ED01D20FC72E6C0D48742"
00562     "77A87B9338291B13CE87478F2218D0089E1D5F2825B7AE7784E2BFD291741B2AD82D201FC7"
00563     "5A487D4F57D6F5A6512C9A29C60E2BABFE00CD3DA6750262FAAE513A9668FF8CD5C245DBD8"
00564     "95390BD494635B46269FC011E2806B06B76DDEAE9E26CBF2A6EEF2193D58F6AC8BC49E101E"
00565     "8677F85C3605939569F16B207BE9DE0D65E884FF8422E82503087811B080E78896301D8044"
00566     "A890AA724D0EC2A58B54D72E506CFD025C0585DDC7BBEF2E233E025DA00A61917DA5045113"
00567     "9325178F711DFAFCD29D567DB48B7CD4ACEC62EDA5AA669557B21EBC97AD815FBB028DAF36"
00568     "976F6E3FF4E4BB6B15DECF164B0193499E3FF2B58A44B5EAD16398834894335F17EFDEB579"
00569     "988CB3CA9D1CD7940F5365A3B71EE7B2DBFC35B853FD8D28BA3759D1CE37542288DC910D40"
00570     "6D2A8DB060455495E23DFB013E0E4AE23576ED57AF49672D07A14C73B7E528ACB4CB3BA325"
00571     "748ED135DA14E1EA838BF54CAD0783772E0651C8B9D903A0F22D0E077F38D74FA47AD64BDC"
00572     "DEC92A9401F780053C0420B21CAFA92F95923E2764E0B5F70AE029B90FA6D1315C7DBCC53E"
00573     "944B62FCCAE0E32AD55BA279CB8DB58BE2B18EF82EBA94636654592CD8C8FD4E5CCC6325B2"
00574     "F11CF8475D7A1721FB52E9E925F05C1CE3DE1D0BF7A73C291EBD881EAC6B771C3EE9FFE17D"
00575     "2372F18CA98627D166B7F280A7DD20C634AA715913ED551E049DA98BF4871567C295BB1653"
```

```
00576     "EFF925400A1694ADE4F681E307F009CF2097D45B3B92910376FDEF17E58E02B67B06C7D5C6"
00577     "29206F03392EEB69349C6178A2AB8C575FFBFE6CEF62DB14AC1E1E9B0C3D9FA63686F3F7B6"
00578     "F69E9B08C406C24EDF2FF5DDA3CD2CA735C4F7610D734A914A6FB8934B8A3FD5887808D236"
00579     "406E3A187DFB593004CCB6C8E74F0C3CE8F09B65C8525BDF0D2B5C22593DF8D00D7F907B28"
00580     "164D1BB312120745DBD623909060C5D0E0EDA4BA7994E5D0AA095506A0E8BE8FC3BD06B37E"
00581     "4CE83A5BDC37C269F035D3DD904C7870FAC049D777511CBCD4A3E13B896A47A5187C410397"
00582     "D1DAD5350242D8CEB657BB087B7AABEAB40F5355C8",  // PRN 15
00583     "C1079AE9F30B67267EFCAB6BA16F939BDE6B605F19B8E81F4F7A826C194AAD2C121735FD17"
00584     "8A0128C4D56398748AB280B71BADBFB5A4B721B0B252D678BD601EEB565E4574283CB497AA"
00585     "18D06E125D31B480F1E9A0EE3E1A5923B27C62884C7A54FCFCF52BC29F6A0F05B1E08ACCB8"
00586     "629D55CABB677B61FCB4DFC3EACDDAA6CFE205E8EFD5EC90B0D91EF96942CB656BD0CCCBED"
00587     "451CA9B5F48708FCED5625E3A78DE96D8B29DFA1EA3854FA08049B5558100CF396FB8895ED"
00588     "D66C45CB005A9C9A9DFDA3CF145B9DDCF4F1CBD2837B5394E43207BF3DFD0D7EDDCABCA69A"
00589     "6D34107E6CA7CFAD0F9C0CD2A9FFBEE9DB853EDC98082744FA41824952D3D0E246FC17A409"
00590     "64D3C1A28D5206AA02D7275DDFFC3A47EA05340E52E8DF0E49CF44B831A5445492D6B21325"
00591     "DEA58B6D702CB5C08562FE3337B5FDDECF4FC921A2D10C3482A7E3725D58D0DC03FF80D03A"
00592     "3B07199D1B3F745F59CCB60271B74B8A83407BA0AAC982E1796CF0F06FDFCC87536AA2F411"
00593     "EC474FEE06A73C2D1B1D6D1AE56A41A87FB5E17A6A2CA02E8C95AC278469603D1DC5842801"
00594     "A8C778FE24D1290D84B932AF7E392270C3DDA8CA9BFC672C45D340D4A19E7CE329E0232410"
00595     "345C328A9032EB2536608DBAE21615FEA83387C4E40BBDABB51C66965101FA6CAB537FE2C6"
00596     "DE4658493200F5FCFE3BC42AE050AB039EF6F7DA5B8BFDE31FC27725AA90A5477684D72184"
00597     "A7F2040410C7492B6C8ECCF95896B0AA4517F269D6E890387377AAB9C2ECCA9D84B5A93808"
00598     "30B40639D4590B37537D2FCD5AD78622DCBFAF60C98F116B36688B80F8BDFE7BA6F6837DE2"
00599     "34F5E68E6B052A2355DD666983E41346B341D427CF897F55182CDCC810A50CC9E2D6030F10"
00600     "45FC2AC7BD0589C552960487887B41AA6F7B856B57EAF114BF46F2E6D9E41A49C33622FF81"
00601     "F0FC4816AA9FACC4DB8CE1E4A3299376CCFF374506BC9993ECA71FA9A779AC5645AC7CC6E6"
00602     "6CB5F76DD2BB922D9CC050C74A1F0E8DB81C72A48E1113F2DE3440DBAA7AF07D78E01FD920"
00603     "29EAF04EED1736D48112273CBCC6F6A07DFB669A33CB169B48614CB7FD22DF040A3AB992FE"
00604     "D2C91F8FFCF85E8D805E152E975B4117AB5A95B8FDFC990A2BA5491784CC4959EEEF34D697"
00605     "D1A9A6AFF427BDAB376B0CF81B8025D0F79F7D83ED9C0F9A9908344A430E9ADDC4785B810F"
00606     "27A8B1BC815B440F1450FA91E455C8C5F177F68A0A614040D85F4833BCEEE54F7C1B1C3288"
00607     "F874B20D9612CFBFF9A03DC6EA85EFC9D79F375A831DEB7366920C2D30AF74B76B5A6C8FF1"
00608     "2946CA1020DC11A918659BEE80C31B6CD73EEAC84E076F5A0D6E8110F22AFA09317D95F385"
00609     "81CC3899A8BEB7AE341395D75BD7B766434EF2794DE2696DE16AD85C3BC368178D6A8C6E7A"
00610     "B3EE93E016025B1369068B37D0C5ECE95FA40913F7D01DBE4C131148E5F4EC008C521CCE41"
00611     "35568BB78FB3D5DB3A99E42164719544FD9C46B81C571BEE4FA275B9177BAF561954383636"
00612     "1D791853D5B85754AEBDF2CD675701B05BA4E433441C578ECA3D0B6AFEC7A25FB7E999A5CC"
00613     "943F856445C2D90026C84A438F344479EB8012E66656B29EA6F694B7329639D1C7D27D5C0E"
00614     "7226C2C060B1EF7DCEDCA203C7690A6547C1BD8A20497C6607097C916E1F589263D6114342"
00615     "CE2631DB041DA3FC4E25696EFD1302F7C9A6F534AD67119FF97D450D4BFB5EF2B171233861"
00616     "768BC74798F325569F40A19434964D7F6FA9A24BF66C066C1E9D05BD00FCA9E800ED8EF3CA"
00617     "C84018670B30AF2A0BEE42CBD4B86C8A2DE436D568",  // PRN 16
00618     "2D9BC6C52B03EDB2B764893D7286EAFE28A8970D4D2A04A8931352C0F75A55268E86B8CDE2"
00619     "06811EDCA7FDCD7D25D4A6D2E2FE5104CDB07C8D38C36FE83E224192A909E46F34B5D19868"
00620     "71B1E3DBA923076FB2839BD4B126F9D0FDF2075A9FD1ED319F2C27DAB014C15F7F5685C600"
00621     "E34C8EA6C6D2F6BDC884E87E9BE5F9551FBD0A77662F323894E2C257F8179A5A9E04DC6E07"
00622     "42DE45552864C918CA4EBECC4692D9EE6777B27C2C0B5F20CB9A89EC574224E1B555B851B1"
00623     "2835967AE9AC982C36161E2E7D0BC06B6F28BB4A631C7163A6D6BF76CC9FB6308A2C171A27"
00624     "064C08EAD35F33A019C1D8E98C50B18F53E1D3ADE72B5E85ED8715A38D70F67BCE396BD8B7"
00625     "EBADA1F957C69832AA90BC7DAEFAE9F993EAD3856CB2E8B9D8F62562F8DB47AFB49DBB0D62"
00626     "DF249887ED4F3BCC175CC83465186EAC4B4AFBE9AB509A5F4EEE58885B913E9E8B77AC1DF2"
00627     "931782377B716EE943062C07F25E65EFFF92C43BF6384B2B2BB893455322DF538E6EA00ED5"
00628     "1458135D0FC5FBE383E22A0BD4538E73A2E973135CA1E3CBD6EF4B07CB2FD40F927CA58A64"
00629     "13543EDA572D3E4BE20A662D22A4DB5FADA3D8216B4DBF0A3F0A20FC6BDC18AAB99A60E55A"
00630     "12508B34E33D1DEC21239B011780C5F3D72EBE627F3B7C8743E4A1AB2871CC8DD853193232"
00631     "A0C55236F22891BF8E50C5C2B357ABDE8F342F9AF1A6E348DFFFD3445EA996ED3D4A889F3A"
00632     "09EC3015472E728E32D82150467FB03F746CFD1A9EAEF91B3E16CBB18A49F21B9D6234DB77"
00633     "65AD1975D77A4CB623374E71435DFB5AF08B6656EACDE013DD1DBB09F4F2D18F79E484C38E"
00634     "1B30E9E20370857B33DF13CC3B93531A0FD8AF033C8431985FE58C3BA62AF728E826A2671C"
00635     "1A63F43DE9BCE537865BB0AF57F90767AD3384B2D4939E026DF389AB6A04C93A1B313872AB"
00636     "554BAF4F151EA00E752914C3520308814C2A7E92CC5F6CD7BD715BC9A143A39BBCD4123287"
00637     "CF8C4ED6D1BCC5F00713C04EE851355BF69C65DCA8C5CB86D46F90AF25A07123C2CC0A7042"
00638     "40257C933B94172B0432E71B489C65F6E9A2BFCBCB7F3B34C28695F925A534D0ABBE7471D6"
00639     "875A118B36D37A298D9AC9219703D184592FC901148F1CEDAA7B02621A4451345089FB02DA"
00640     "94D8F20CC8EE0E0924B041E5FA4774996295 36E3B33693A5152CE357512844E5A13029485A"
00641     "A4169F53ECC1329C2B844ED50B32B2A220613B1F42EA888B68383DC4741B154F851A820151"
00642     "A36C17621081096AD3D70BFA8847515989CCAAFB3CE06776D1DB548693322C0908E5B063A0"
00643     "D482EF1EBEB75861F2B5A8D1FA764BA121D69DDF1231E769E095EF7B42E38EEADB19F7E3BA"
00644     "CAA9163B9595B71A653CD0296E34E3B31F591A1EEA947AA36C55BDC958C8E5D499697DA820"
00645     "F8D9CDC09B751E3FDD3EE49FB34E9073C08682D1DC7138E2A2FAE8DC4ABC3A28DCB11A6B9F"
00646     "92E2A9881CB731005394590DACC935F572BAB67034B6D9CF0E966B7EE7A3379DFE33C7D0B6"
00647     "21817B8FA9A8D93EEC77BA493F8552EABBE2A6904C7592E206A9A152204EE63021CA7555A0"
00648     "5DDB27FD8FC70DE982BC7E5838CC3DF5ACF7F32BD70E65BDF4B7C67901186243 73F532C7AB"
00649     "82713B4BEFCAFF2EB55CB00BE91CC69DDD1C5FA977782609200BB785E86FCFE21D17DB9417"
00650     "FCB938D0833EA6BD96FA1229AEE293AD0880E3DD287371CEC3F899BCFA9DC9B66FC627307B"
00651     "A331F1020AF6233EE00FE8B01FFAC9CB7C464F20D97418C2AEA800C878551C701CFB3A9645"
00652     "FB70F2CBB8C0CC73F6AFE69460E6D3D630EC403034",  // PRN 17
00653     "BC514690938DAC703824A90124E422A0A9B4ED89CD999D8B2773ADFB66A47505F8D7B6E848"
00654     "79EF37ACC61C346C0BCD5ACF4489A07956CDED10B81CC62EB6E8ADF7F4BB0FBEE539BD6FCC"
00655     "B4A4DF88BBE3F8FA22699966556731BFAC901215DC15D71F3281153939D05F90D79B0FAEB8"
00656     "EDFC15A2A788CE9A405EB431E311AC5650CA6E3B865DD0ED525448DC11BC7E77668EB1A601"
00657     "5351CDA4F46EB948FBC9C0299C196745094B023F49A07C4CCAAFCB298FDF9F0E73BED2410B"
00658     "E20192ED22136F4022E37E6FF7A4399C609CB2F5B51031DC4BEA90E4C379CF0DEC7C123D2F"
00659     "2D49CCB35D0E41057CA75A6D7063E7B50B2CA0C4C1163745F143201D393EF90DF677A3EF76"
00660     "1A4A9BE62A15C51AE02381BF1FFAAE4933DEDDF2B64F36CCED0F218CC3A3083B81740FC97C"
00661     "E1D5DE2CFFF344969E5A670AD3EE658AF9427B9D738A9110BC0B45421C857FAFE5B8FDD1E6"
00662     "C876D79F451A8920BDF48EAA982D7FD91401D51F0FD514A44568E2BA8102956532FD3B3B64"
```

```
00663     "300FFF16A23AB982BA7AF15C99892E4D4C8FEBC79BF5DA567DB0A7C9E781BC5B5D7089056C"
00664     "13D1BF8B7D18C8DF15D223F7071B641300818B828A95CE706C7B40FF21DB74474D6412B6AE"
00665     "F302EB0679441ECA16AA4A5FFAAC5A4BFD8E798D8790711B0F785386E45BD78B010D719FFC"
00666     "32DA0465EA087F6837A3E67C33B9D266683500880F191154C3ABA78944805D4DEA155A1171"
00667     "2B72321C62591C22C3E8EA9CA1C55E1B2C10AF545C7929A8F035F5C03DBB6B86983CCC37CA"
00668     "59D58BF0204BB8D8101714F7D5EB26D19331CAB21A24A8A655B18798D7C3302168408C775A"
00669     "B5243DE216B0A7FF61ED51D4F2CC30B5631EDBB84673A1FD3C17006681F05DF6170213AAFA"
00670     "423EC62761E64821758E9FA8696471298325E8B691D9FACE7CD8E05FD6B6325BE098123BDD"
00671     "0EA59BA74369A87270675C05CA79CEC1572647DDE6DBBB64AE43A760561B1718124AD77A5B"
00672     "07CDEFF96D436E208A9D84FC6FB001EB30789FD5F1326A33C9F2D73175D9D6E4F02563A59E"
00673     "A69490153783A84C2955BC1CCDC0251E041481110D91212CD9582D6D345CDCFF2C71B08CB9"
00674     "D3480C65619BD08112602329C5F6D2652F5FA770F47D0F61E7ECF2ED8205A71B6667181B5F"
00675     "79B93D059469A71D2DA196FD98690BB58B4DC218B16DC650CBDC57FF7FF38909197C597BCB"
00676     "DFB7ED0F883983C18CAFBD80D56C071E2C2B64E33CEC03A58225AF13F3A8517DF3A23D90ED"
00677     "43E7F429E5EED8F99F136944B61816E6B9E147B173AC33551C4B6C05FAED10FEFB202F4EBB"
00678     "0F4A73AE53D3E7600F268D557555CB61A823D36ED537AE76FB96462D8B5ACCE8427DF32371"
00679     "FF4F32E66896D3ED4478943F08D0B7BB528C2A39E7CF77C621577C3DCFFA5BBDC6A4D3D0C4"
00680     "1DF0A7F141EB60B63750669ECE2CEC7C6EDF0660BF96749D01727040AD51D4869585812D3E"
00681     "604F94FB8C90925A4AD3859D7DFBE83CD1A88F8E3F15EA96A4D745301B3480A2C6463F885A"
00682     "32BC173FBDCDF80CC8BF8768DF2041ECE73D9F17B71119438DF45B99484859BEBD1938CDC0"
00683     "CBA05B9E117BB5F2F205FE6B50B602B429C8133FE52FDD887443EFFA8176EAA21FD567D124"
00684     "D7912599DF30E3B760795724364391 92C714F45DEAAA892256DF38F16825575F9AAE7FB723"
00685     "6D1F8860071AF421D5F6F9E75F5D39218726E24C8BB3C9EBF3269579C421DFAC0B5D369103"
00686     "1F444B9A4E7004A91A6061B3C0CCDE29FBF07B1A0ED82BC3947ADCF3D87B423CA1694310E2"
00687     "9DD31663C11B8B8E062A9EC4D4E0FC19DF916A6C80",  // PRN 18
00688     "F848B06D24B8741482555C3315EC3D051B23BC975819C7A3B0D152E1970E1FABF323853BF3"
00689     "7008713374D685E9CD2CD0CEBC6A8D5CCD62FD5603BE5074AADF79C6310542F1747C5C7A1C"
00690     "260AA8FC4729AF5E497DB117D635DEEC76810DDC5E67BAABB45F835D6580075744E36BBAC8"
00691     "0CECB538A8EB36F60C3B47821CE2EC482CFB80058E1AE5CC37700AF07895D30745AFE02235"
00692     "B1E7ABEEA172D60C59FE0B386A615121FB1A4EDA4B4A089EF0B5C6F273E3592DC84E3499A7"
00693     "89903BCD8B7476870F2A247C5069EB644B607732089818618BE394F3F99FB340828D0396B7"
00694     "807E5F323884023D371F650D95084D2C44A82507509497A99EFE83B4C1F6546D657CD72DFC"
00695     "2A6206A4C44D1DA9AB81EEE49DFEA323EEC5FF3BB4CC02F34AABB39975E152327682BA5EF0"
00696     "EE8286367D49F8C24496690189DCA21361372DC48BBB3DEE1CD96A9833DAD1A161838F7E63"
00697     "BFB67A7A5740D7D7C15051EF9F1902BE1806DA5207F4F1715ECFAAE928035C5661B76B5703"
00698     "D5B9444D35FE966C3553918D525579DD87ED0D695740103AE9B49E034C5441DDB8D68274DE"
00699     "9140839AAF70A2507E725AA9C1B7BEA38E45C85CE08004B2F96B5BA4181E4F8B53DF62A12F"
00700     "D44800AB28076BD11FE30DD46AF779DEFD25089104156F9B4F00FF4374E6D5B3F40470F8CE"
00701     "75B2C59B9D2B8328714F446078ED2CBA3AD5C34BFD3B4D673B90EB66A67B6BB6C92E5BAE22"
00702     "A663A7C4A8E279F5604BF2454E8CD43154B68AD5A352087CF5503D43AD65C942B2A24BBFBE"
00703     "048ED195E75831E58F468FE3049B3BB0A53908066A6BB912ED2F76707DC5AECFA0D94C929F"
00704     "1EA950F70163F0373A1C450F62C013AA72FD63ACA6D3C2F3C1DCB571FCA6818F7784D8FCBC"
00705     "9A3CACA2EA577356AA03497C4811E5418A0A7FC97F01D324BC5BF50025582D33F3E7BE235C"
00706     "717FE3F8677664800AABFC0F846998CADF955B7F039171A9AB4A5F06FA6FA682F4989B0088"
00707     "87F46792102BF23E66ED196C0033E225444664734B62FF55B277B531FAB8968CECCAD65E22"
00708     "4CFF621CEBF558D3013E6F7545FC58FC8942A469F32433218725408122 11B8EE2C980138AF"
00709     "4CADC65EF515F80370BDBD8A5DDE64A7840FDBA73BEEF284A2B6260BB668703D442EEBC1E0"
00710     "67649DE66BCDE78940D8B1CBF30B431A53DA99E635AFB5617D72CC4A3F8E55C428D5D7D3B8"
00711     "E4E152D61460DF112531A5D9CB7C06152E893088D1C5EC150991E46C8D310B3954E5FE200F"
00712     "B74116E9E37AD62044FA95A011F1D4427984734375E05C46BEF6B65CC59D45BF74FFDE18A0"
00713     "37D34A4A084BB356029431842EC8EF5F0485E32F0898DD90BCEB47F1782B02EA1C580E5C95"
00714     "C7779A0FF2A2807A05D62B9A1446F287569E60D73A9695A7B6CEB34813B2EB456239567F65"
00715     "7A6CE18DA28485850D5ADB6F1756948D4BD4E803574A12DDB47F5D2D164713F3CC4C60FFAF"
00716     "0B5ECC82AED6F19916F0CD8C1452B41579A8A40E7A1D4EF220E6D78EDC6A592E2F30CA677A"
00717     "F87BF22707824D597541CE0B44ECF15683088FA2ED3CFEE9D3FA26A3A4D405926511B25909"
00718     "53820D4DF802D00431F2AC3FC25AB5684018BFD1F5D7D714E5CEDF0F0DF6DCBA6D4B32D88B"
00719     "B1B740608CDE52DE67246AAEA7FA0111854B8DA2BD0B923EDFBA37330144DA966EBCCC5860"
00720     "0724B2EF3BBA076E5B932B34F4E4F4425E2AFA63F57D932545299FC145F004C076E8BB4D37"
00721     "2F73F6EB6653F779456FFE3A3A12760DC02CD58B5B891436406A7919DFB9AEDC7D8A82F4C5"
00722     "85483982DE2B086258474302E7EB26DA4183D363D0",  // PRN 19
00723     "4F01E8C687252CAAE1EA80DF4411F92015BDD7DE90F317908FABE7B15AF8F7DCE47FB3B3C0"
00724     "9DF31F58C41FF10A2748B2DE8321E8726F5FD656B7EF7F3154EA9A92E459551913DB2A69D2"
00725     "AB35CD692422229D5D7E40D18C7EE9B2560C7A9972D382C41C3C0821FF1F25DC86C8CA9F94"
00726     "E2054CAC07A3E0DE5CD84127C2455586A136884E15573E8711BB8B250C91AF5F78D8701E014"
00727     "F0D3B52951AA4EC7BE99F4DEEA5DA94227665E542108C2D674ED14A129031F6F6B465D5CCB"
00728     "450491F555FD32E89909C88C3CA00FC6D2AA72E6E13C6C5590546C8A75D8C4A14F3C5B4194"
00729     "D23F0DFC0B0A59AF83382A9630F81AD5CF275E1703EF17BB96D0F678396C898F462BE67793"
00730     "0D60CDB0423D8AD9D5402433C8CC3C2AE542DC3EF514DD880DF842082F2074249F07AE3EA1"
00731     "D4075A87CC2F6F869A216D099309182524A528C2E3C472A738F2061C87C93E4C0AAC9760CD"
00732     "0F837F0F2FE9624FDD46E58A102A3254B4B52A315E09B0D5017BF456741E542084E861B882"
00733     "8E7360F988106F258BD99FE5A89B61D399771D2A9CEA27747913CC00DE32E749180E4DBFB3"
00734     "E7F7AC12F0ADBC02BFD1CD116E9628664318513774592CBA8AFE0C0FE31384C7453C07A40D"
00735     "AC6BA47BFD19BC91ED5E808DDF9664567A99DCBA919045BB79B68CC9037EE8C5142B72B65B"
00736     "A2E8671D771D91BAFE5864D63FC72DECD8ECC441C03BCD96621BCA1CA20A81A4CBE03F20F3"
00737     "259F49835062F42C247364ECB33471F0D3316E911B35F80A6C8FF8B5C0938F34B1DCECA59D"
00738     "2F3F44E32CF776A3BEF5517010CFCAFCEE3AF65029C3B3D97C2857CD384DFBF2AFC9A42829"
00739     "AA62DED01318CE940C35E7EF58027FBBF795833285BAB3CE220DE17CC17B10BCF64F53ABA6"
00740     "4016830445C7C4B483B469640E2FC686FA23BE89B91C28B3D6750FAD9DAAEC3E654B339358"
00741     "A474F49A1199DB9BA703BEE4ACB98A95F16A86CF5044F0AF81F5B8DAC4BF807DACA999FF3C"
00742     "774A2FAD89CF07A726F9854CAC24F3437D03AEE6DEF8F54642BAEDD8998D5B28743F715E51"
00743     "5C2998CDD3C880323A1EB0728260B0817E0FD0C8D9E8C42B43245E43A2C18131476F78EC78"
00744     "A8BE7EB3142BA8D4D66EADF2A61B7E0BCEFE26CD1FF44E7E796E0A917E4A71EE2746C6FE2B"
00745     "F313B4FA3AC922327D9F819A9E8E15BAAA7506C1CE653FD4D4BEA49191C21157CF60A9EE03"
00746     "A526AD9E954805E36B5D731D5A23B73B9974390C37A681483EFB649734AB686A49B51856A2"
00747     "C8F22E5B5ADB851C7448A98013E7D106D153A910AA814B2E28084B6E71140B30F39E1BF1ED"
00748     "F253476215F80485FB5D7E57AB816F956D61EFB7D6BF98EB08593911D2A81D55FEF4382DA7"
00749     "B4BF2D3F5ACE9B1F7C749CA05C3BABCB361DDB89DD5A1B5DB57755AB8FB6570E98C69E0F14"
```

```
00750     "2BBFD5E505AA06D87210E2439D2E9CA9DE0952B093893DEE2A94E5BC77B65AC8C713223097"
00751     "87A70A5D64B0858C6D137F78E5B0FE02A56F75B6B69C24E87A3DA1A35280EABDEC95E8B158"
00752     "4889B6ABC004F51FE5631D59FCBFA156D40805B3A5F5F68AF687A81E17CDD77F8ECFF3CC86"
00753     "99B379D292A9ABF8A3359BA6A32E7D0261720B8A3A836B081FBB0637B5AF6CAB7862A1E8CF"
00754     "1C1092924733D530C38D4F77C842D2B0DC97AC52A832B124BE512E336D743FAAC4AF92942C"
00755     "CCD61532596809D420860847D5E270FCADD45521C34481EF5AAD4D025E63ECE511363D9941"
00756     "5A6D8FAA80CCEBB2E0C8A949A0D5C415F0722817EE147828551B7ABA696155B76632EA94AA"
00757     "FFAA69C5398BB67D6754B6BFB7A351631CA179C1AC",  // PRN 20
00758     "B16C9B01A00C67FB6F3E732B0B428D301E56C43EED81DBB504B4DDD2A7E4B3180340DCC6B2"
00759     "D457C56DCB8A98E1BBC98FC453EBD696E9D640B8FAF7EEFEA4B78308290B5C6B643F97D02B"
00760     "689A0FAE44BBC9F8DDDD69D81D58A85B5909520140357B0823E72C7CF345CA0ADD88FF63E0"
00761     "049F206F03103B3D402DA4E719C0946B882B551B595A1771BA29BF91651D54BB918475A71F"
00762     "91D11D9D2B054E312F1E243A24F4867B599CA4EC59D88789D00381D872B5BB14CF44099927"
00763     "78C9B0ECF170076D1D28EA18DEB86898A5278FB285415C591D4A694B5C5E34C766C9BBF147"
00764     "F6E780F7CEE9F47B411C1BAA8F2B669DD1DEEEA817D2A61A406D97E030BA1F6E9B2A9485F6"
00765     "E1F30185790895FC968EFE0C274817E860E2ED408A6F34DAE84B895B19F0D75A961C993781"
00766     "D3615762707425537717A28ADEC678394E9182543C5914593CC12CA6101DA188A0C3496398"
00767     "28DC5480FAF82516CCC6FD622F1DD7B50F64ACBDB8EF5CD91C2482393A5C72A9BC07388282"
00768     "D393FBAAE9C0464ACA0BA08FADE18963CFD2948DA2D9502E759432145D3278987243117EED4"
00769     "82B04EC24613979DDAEAAF0EC1C4AE757494F06138613F92266C9C38F063B8C6277E486DD8"
00770     "DE8C01B5EA87BBE8C088FAADBE209DCD45C8E9AF54C2C9485F7804866E4A9F42096E54E272"
00771     "8F691DB707E3A129A22C2C4D4726B8CD13C9924F388070200BC186E3F0DB92195609A064A8"
00772     "13CA6BAC734C35B1F3AC473B733795F6ABFE0BFD79A046102394AC1C1C4B7EBFBB26ADDCD3"
00773     "59B9BB151CE1AB86DC5BDF05723644039D8BB1A7395789A0D10F7466788106187A8A93F388"
00774     "763FAE0A33D77979B59B24CA8B8DD62A7FD57E1F11003B3A8E0F30387C6CE86B7B1C06DBF9"
00775     "E1881EDC0AFDF04915E1E54A392396893D3AD7A56464AA860B61BD7D7C144DB048B959EFD2"
00776     "8B7C5A1E84EA603C415103C7F418C919EBFDFC5180CC0EE39266D57EF24B335F4E2B5FC3A7"
00777     "B46A8C2A814A03775DE6462ECF0B6EDC8299AE153F7784B6008344D3B0AF4A4B3D721ADF48"
00778     "56A5B96AC67BB26092FDCA5793DEE80ECDCE63EF4B62A3035477F8D27211EFEEA68949F1A7"
00779     "2F88BBD048A6E1D19D5663CBC851AE96F090B10C40126043952F477F7E025157F07524955C"
00780     "7B9EE83F5375BE3951CD9858C34FEEBD8128520BB24111C858ABA3F2EC2F0C180925C2EA9B"
00781     "FE80B9CF140CEFF4199BB6870730CDE78954525ADF7F97002AE2C56A098FF8C7FBA7C8191D"
00782     "7D5E923B206145C8619D4B26C587A15C00C595DC692BF9E1E9852C9B778575E189636B85CC"
00783     "29C63B49ABFD6BAD698F89EC4EA795BD54C2048FB687E283495617B90F925FC8B6FA71B73D"
00784     "E63C4E00777B41C3C255E3D42570AD4568FF670F864D2E2BD0D906CF2F97995BC669723839"
00785     "1BE458711123E656E153B638C02D565FEB6F8734E5AB231D7A3AB3A8DF368011093AFC5062"
00786     "5C27B6545EE7D8E6E708D7A0AFFAC25E27B6691EB3C7AC0C661555A151458C91697DF976F0"
00787     "00828F39FB42FAEC5C7CFF5BCBDAB1B48EAFF35132F3E89567954981D617B000991644159F"
00788     "41BE834EFA25053EF6EEB51BBE36BBCD1B3E5AD9FFB24707E6E208979E8128B7FE97B7C99B"
00789     "E82CD130ADE9061EFB11069B32CE0A165717475B1874A4873C29A396A983B0EF949B65E268"
00790     "02D13B1F0503617C9E1CB6050A6AD6202DB54DD94C5941485FAE5575B03E445EFB80DE5245"
00791     "6C373D9C40DFBF9EBD4172F75CBB72B4FBDE8B57A768C3D723DA35EC6D79FF48F94211D431"
00792     "02CAADCEC758748447AAD12C61C651316C23C99DE8",  // PRN 21
00793     "B2827D682D4012B8F6E4C6D5C7A047D69B1A34FC25715BBD5CA1CF71F88DFC65871D7E9D49"
00794     "209D1748F1B48566AC6CB8D253DF76E027090F5FDFC8A0EB82B4AAAFB9393F13F92A7902E2"
00795     "9318A5A23A4443480419A0173F846AFAE964C7E7CAE99229D6821DFEFDA8B195EFB3176D07"
00796     "9F11507DF23FD0C261EC61A0551136F7322620E671BEA5585A092DC613F79BA89D38BBDCB1"
00797     "73949A580FE9634C2422C609C3EFF952C04CF103D713544E592BDEAB6DF35826CAEC60F24A"
00798     "A07F8B463F3F09F9967060DB17AA5B86BFC695F20E0C16B8500A68175E7671E442E83764B1"
00799     "CC114C90855E5B02474A24824AF91B377C311CE30B0CC39F934C3F659A6B7053C9DAE76C21"
00800     "8511E0E3B94CC15ADA4FECBAB803DCC7D6FD57D22EF851640B050C3C6977170E7290F0B02D"
00801     "B760E7A1E4868D1E0C2691A514590DB206AB6BB868D4D8DD26D3D769EAA0A9504488611D8B"
00802     "63D03ACF5A6DBA6CD79724B656F8D02F7317822B25B292B202ECE2F841AA007FC9B740D746"
00803     "DD1D20C9046C28AB6B8A1D6424A096FD4D2AE871ED0FA56F544AD760D798F167DCE392C184"
00804     "BCA729C4A10D98BCE22E7671C93D2A6A16679E8F09F961443A9C09F08AF2F8573A0FCE4F3F"
00805     "2C429C3AA5379E5D9FD515A6F4A270719B036D6857FBDEF2485DC2C9E55F4530B65B090799"
00806     "D4A9D912F824414110C28D946A5EC3F950EEF9C4C19BBE28B5F539B50DC2A6D5102243815C"
00807     "68396972A7E8C174B1909E38FB010805648137A48B71FBACAA5FA2CA57BB6166897EE52EEB"
00808     "55B12441F7FDDBE24DAA6F3FE75ADF97BDA227E7755B3AFEB342ABB64ECE740C3ECBF59DCD"
00809     "36BB735931A7ACC963CFF4E694C4A501994082C5F68E756818AC2D919B8476CD933C110DDE"
00810     "2172FB4441EBCE3DD21BDDC3B0EDE857D7DE44ECD00EB5ACD8B6847F0621EC24357054FCFF"
00811     "57E66B3D1B0ADA0F74CBA7D510F70062980E7B0E0542CFD044ADD438A0581113CDF941C9F6"
00812     "840F41D91935B53FE7FEF3F163BDEEB77A706321E6F18DD1B7457D387B2EFD40A5712C58EB"
00813     "A2BDEA94C738907B11A5C5150BE6EEE643D8A1EA1A0A565D559ADBDDD3D872BCF8FBB40452"
00814     "5F791B5AF7EF3CFB6FE10E833F19AA3F7B48CA9602212959FE244F3D3B2B95673EC9CB8EA6"
00815     "D356125F4A6F8232D39E95B80F1C057CE08E1BBAA8569A0F030227BA8E5673BEF1B5FF19FA"
00816     "80A8C2AC57A1953729881BAAC48ACF026D9F8B7D4715E16A14A842F7BDF027CFAC22C3597F"
00817     "81A065E985AA225EC7F0EA603995973212859B1072E081577C2EF6F6E883EE7F495BBFEC99"
00818     "0479018457F50DF81CA020D3C7F34A0152684A31E19848A1C7E434B93E4221A77D39061430"
00819     "FFF183D86F41FF2F1CC967993701737F71CFFDCCEB0EBED8B8207856B514393AE1709DF732A"
00820     "55983E9FA53AE9AAC855B70762F9CCEB49A14DBB200A4E612D0479DE1B09DEB4051C934920"
00821     "F0BFC6720FBEFDC6892E37D8AD3E46C1B41C8CD1F85B833068E70E0DA5054FFC9A4B87D49A"
00822     "2B082A201D0F67FCC2A697B4C5E03BF826FDA853C7A7096906B3F169472141C3B355D25F33"
00823     "5C4FF43596F6B1A7084B173CC50D68F393883673BBEC015A65D9A729D92EA8B0369B465751"
00824     "30E9FA4C997730B7A8A8B77824DBFE94A58BAA56E798C4AA866366CBB569751C89658280E7"
00825     "ECF0136BF6592F3BC89D2FFEDFC0504F9AF6F563ADED00D7307526283675E341A8A3CED3BF"
00826     "371E488DA987BA3C343B5D06F3A1D41E1B61C86A9CD85DA34A2119477525915059 5D9FE6E3"
00827     "5B705CCAACE25DE1E669AC6F7D9DB5E59537978CB4",  // PRN 22
00828     "16C80914164E3DF298C8C6BC874DDA6804D47E82B95B17BB0FEEEF153E7CC64E5AB8ED6A4C"
00829     "7485EAB7E4D536EF3FF2FD6BD69C163B0CC41AE60D8B4A21E1B0B7754E075B642937B635FE"
00830     "733D47FDEDBA00CB8E9739B47ABD1EE0029E962EAEBA5268F22CF02B638EA8EC73444AA98C"
00831     "909B19A5A8C5C6BBA58BAF96AB42055799F93D71D631B708E4EF134BD59390F84FEE606DDD"
00832     "04C106C612C075FD31A5542DA5D3CC90F4CDE79942D1CECE5B93DAFBE7B51065D3C78C87A7"
00833     "38B11C2961CD0555E821EB7FECFF00595B50AC4EC507B0FB01013DB5800CD404A53B99AD74"
00834     "0EDCCC182C12C28062271A15DCA67708091455CDC07105029982DE6A1AC2B94B1AD5BFA422"
00835     "E496C906E9D63738BC10798453154CB20525FEC9B7F06541FC9AF94678C6DF98226CD62635"
00836     "6BFFA68ADA72BBBDBF8E626F1B6DF5545B865103F402E54DE75120850F0C4230B423E90096"
```

```
00837     "D517E19787AEF96FAC40363469C5D5DF8AEF1D86CF30D10FC805F7AD52555D3EC3E9E0AB68"
00838     "1310EEF51A2E5922D9740C972C9E531B90F68B4150A39C8E7FDA224256BACB84D0A65D2419"
00839     "87B9B6A4EA6726073C38E374124497FD1FE998FD268D4F0FB6031FDB3E17863D81C418F8BE"
00840     "A50963B565E6FB80DC674F5602CAF63F4C149F5FCBEAA29B36F038FB8CC7C9677190E1F07F"
00841     "D114705B11AEC538A2C59DF951B95D07285D1C67BAE2ADBEA859AE553270C26581C633994D"
00842     "5690BF4A4721E297D1E5B4E1CC2948564801AFBDEDCD3F3715FFCE54B393E9E445A26FAFE2"
00843     "EF1720A05C9C774BEDF251AD010D12BC4417E8FBE9FE17B88811646773834A44C1167BA1C38"
00844     "FD8BE5754D4E233D12DCE77A697970C8BFD3EA11C8DBD2C66AA8D9FAD8AC8D5F891A4AB4EA"
00845     "4335C485358DA5089A3E141C221B0FB0D9614FF15269183ADA3A205942B8EB800964F69C34"
00846     "CC015BC2779BAE3E1F74FEB81020F1B10267C802178DA05C377A433BCE4F2115A0F60621C6"
00847     "23F3435F5D02B38FF97BC3816E6CF29AFE7AC1F00C4D8E0BF15A3EAB418A41A55E9AC6B908"
00848     "E3A894077D6D7515CB342B8502B5F956B4EC258C883E804B451E518E5BB9EFC526CC4155C3"
00849     "ABB9296E8FF8385D74209624E89B6973E82AFEB915AF9E0AF6B48BCFC4666447452F0972F6"
00850     "25909505BFE9E21589C512CC0AD7CCEB52C12B6E4482E69A0C394F453046DCB4E7353AE007"
00851     "6F0D893A988CD9EC6B05E38EA856EB827007BCEBB678CA3DA0737CFA8ACDA7BE04F0690201"
00852     "F43B9DF94900C801958FF8C986ED9A668FF9352F26A59C333A2E9556145C79F9A61DA5DF02"
00853     "9FC931F12A2CB0C1143A04307A97A20025EA277C66D8ED1CC0ACC9112BA9836627AF2088C8"
00854     "322CEBEA70B9AD231EB622300268366A89E0578C6A1343F616EA0D15EC098BE2C93B5A2A1A"
00855     "B5D49C2DEF7A35E708A5B28E6BC6EBBE823FDA837E7253859A5D915F9DD238D8F858471E35"
00856     "BA8C0AE9359FC07258F02BADB9A71665021966783BA3417D19362C6046831B7BB6DEDFEF78"
00857     "C5FEF387743C9AB786E861C3639D2A94FFA920EF4B1C0F1E59CF7E36472ED1DA506847EEB9"
00858     "0AB79D09FBA416D46B8D3B172C088DCA4993C9AB7BB775CFEF639EFC52C8934DF85E598B13"
00859     "60001CF805F99DB49B5F1C2D28049013D4FE693E64A61B1C60D86A5CFB57D35AA8667636B5"
00860     "06D3C1DBFD1C3DCF977E71CB929591311C8C82029105D32169231690EA44A37EB4C66A60CA"
00861     "D5169FF9FB4CF9BACEBD65B13A082636A7437278C916B33CD36BEDCA17A09F603300759DA1"
00862     "78896C84E154CD3A34A527E57685478A99B5199094",  // PRN 23
00863     "7B570F8F888E0526F19DEB03BD80266062810BAD10649EEFD052E9BF70C29151F09A63D86F"
00864     "74B43AC98B1EF5989CFFE4A61B440E01B228C47A89C658503F02C375F5C66DC2EA97E4EB7D"
00865     "D74393F7186B6066C58477C81C53AAF90D62BF28AF85F0BC689852E5A9016D4D1E381D9746"
00866     "E65F26FB0DC9E0D4BCC9030EDB64540EBCD953573649B5CC607080E1305CDB6460D26E9F86"
00867     "A835460057F33438393CF7FA700CA2ADD69DD79432A3EC0F4D1DF72A9BCB7E531323BED411F"
00868     "D0202AD8A2E746C3A28DF68E9B846C8AEFC952EB6720CBC0D23FB503ECA6905AE3661A96DB"
00869     "B09B63982BDACE71FCEDC2D0F7DA43963FBD0674719213176D002CF6F7CF46A77AF805EEE7"
00870     "B98A23C31ACBC9DFDE9F97A57B61BFE84B62A8482E0BB0EB7AF54E05C07C3E2C63F3B9FD46"
00871     "3679ED2DC52198AD3E7439466E7C1C0C5E24D879DBF34A5741FBC03A4C3179802A5780FBBA"
00872     "7E0FBDF99F0EE6E19E926F8E98C1BE65A38ABC45B273CD542CD10EB90C440037E90E97E4A4"
00873     "B66AA84458A94117F78131363D62CB3CB62DC32CB9C5D94B7A565B75D0FB8C8AC1AC5484EF"
00874     "B83F7262B55C19F8083904B70CA777796CBB8EAF62AF899D2CD41DE92B0FC573AAA27C5DEF"
00875     "63FD29961957E4E75F0100731604A611EB59BCFE6B371D58FA92034482AA43A1C88A73FAE9"
00876     "A0880287840FFF447FEA1F469DE526DFAAC6C0672AA3DE10CF9D78BA236A7BEF52BF0B8F15"
00877     "F2A33BF1C4D7ECC16B4C20198475714E620F694637F7A0BB9F29C76404DFC147472BBB8107"
00878     "629C141A7A69CF6BAD24FC7CB321829477541281A0DF4CDCADB1FB1818DADC8CE4123BB413"
00879     "564EA8D98DAE969E3EA57EFD3C1CEF4BA51C001CEA267033904831B96F867508F51094E5CF"
00880     "E6DAB34F7879F57BDDE864DC52EFDF32C73C1E4B26EC22AAC6A9156DE7BE2DFE520F432548"
00881     "8EFFB330E196B283DA0DF20E3E4A1D7A5EC1014254110AE168A22D61AEF83A3D89E5C73CE0"
00882     "A55BB5E55A975FE534D0CB6D2A51143BA2D6E8A10DF83D97A2E6FF40F68B39B72A347C6C1C"
00883     "D4F92493E915CC10790D7B8FB5A4316EE504218B02F2D940953DDEDECBAF9CE150405EB2B1"
00884     "947D2D0045D9FCE83BCD530AE4A75651CDE9C13AA161602545DB39FC747D917F943654FE5D"
00885     "93CDEF3F8E647CE11167B546688FD319E0DC6B38180ADF97E6150DECD83E5BF132809CBC76"
00886     "579E50CF813D94FB6AD3F0EE247DAC86811CF7625966409DC99D907316C3FBE4BC26B8537C"
00887     "C1471846584D710157BE09D7BD6CD46C89C9DAF39DF4EBE657E9035E78E91D94263B0D87B9"
00888     "8C7E4C09F741F2D64D7C84A283E05CEDA7748B74578D1222215A4826D1680B71DA08A57A89"
00889     "CEE3CC46BAB909C21BD69ABB33A2183E254C87DCF59529315DF3A378C8CE26DACAA3DC9DCC"
00890     "3C1572EAE6A3331F369955CD847C801B8B69BD3B28525B738E5B1091668D83165EE76783F2"
00891     "D585F86F88A958F19F2E0378DBF0998E5339B39FED7D8019CF773C6EDD985A1248827C5630"
00892     "2776D5AAADD6A3350D844B766450566DC07774BAE6526C5884BD1C12D8137A2BBDB249D143"
00893     "4EBEDBDE69D770624F07C69E59B765866EA0D2C1920C21C832AF4DE6CA22147E4D35A412D6"
00894     "F5CD1661E47C0C559B4FAC070D422EC14A0561852011ECF5F57C2028876CDD75E2085A36DC"
00895     "463E5C6562D277B10B543DCDC1DAB155DA7DB8294CA040D66C770E6D1551C2CD3D55F42573"
00896     "5055FF22D081EC031CFE8F7B590C3BEC507D160F190B7C0BCF8BC82848776123B08AFB0402"
00897     "3DB0189E0AAE9CF350F6E7A4522BAA5F079300E908",  // PRN 24
00898     "1969C08944104F0F6D3DB64D27C76521D9FE8C35D413E0F30B4764CD45E6847D13A2866144"
00899     "36C0C373D41B8A00240855A32CFC0F2130CF98B5006B86720A71D6D307B64173A2F0ED9C7A"
00900     "3B5F3E824B4290F67DF6479C01DE5D4D785DA63D039791D2F0C914E0BAA93086052E74E30B"
00901     "0D2F667AA09243B0E5E953685C90A09CFE6F9B1C77774B1AD281010BDB5F8B333758F5C607"
00902     "695E005D8C004D2958F079929F1E3098929892B6EBEE479A842FDC51A90B477A79E5023584"
00903     "3880CC570E06C6224B2CF2B221271863E5BC72D2414AA28ADFB349B53BCDE2A2669BCB73F7"
00904     "AAA21812D9C3A22D29345469BA48C31106E87CCBC14EA7D603367CA4D04D7CEA77A9E89AE8"
00905     "9252F6584CDA1157A552D51C23331DEB200590EF5788678578056AEC9D0994040D50D8D46E"
00906     "C70BE649B7818DC31DE449E5BE741319C696CFD99CEDEB776D1D88D7540EB76455DF88262E"
00907     "3D86D7664484BAF9E6EC2BC51EBEB90C3E76E3236D17AA341B3B7F1B3EFF196085B46390C7"
00908     "AD5741751DE655D347BED45207E26785A65FD8F8E1408A9B8B58D0DEBBB73468B08BE76EDC"
00909     "DAA55C61197AD8E05EB70D553A3064A751AECE7CBF769AA5306CA89886AD6789C05B7723CD"
00910     "F88F28127AB5D8905D9D147F77DB1F6749B61625E41B0D81905AD844CFFAF2D7EFE4880B53"
00911     "7A0C2C35090EDAB69017C443A0BD36BD0138BDA4C7A1B1E54C8D57416C619F6CAB2144D681"
00912     "F3EAF9FB05A6FAA2FF108940DDD5D0716EDAE7D004AB7AF21073B76335D2AEEA54A8660F14"
00913     "D091E46603A748B4D9F3B66E1FEEF2383745FEA9ADDCD12A12C66655669BE7023A01DEB969"
00914     "3B7FAF5631FF3C5A91BAA85F7DB720E6189EC9B068066D05A6CAE245D6C107993000D8BE7C"
00915     "32A9B659A02C9F74471196A3EE4774685BFA93F1C2286B94BDFEF6B653A2520ADA1DCE8A6C"
00916     "6832DF79F46DA068A5DD6A40587CC8A928690EF9CDB73FD0BDAC4A0B790309B6A27FD20F23"
00917     "AE7877E57BF70BAFB9AD6E50DC4062B2CD393B8A456CD56FD4FAA1D42AD33965ABEF471123"
00918     "486DD9ABBD8DC0B6E74DB20257D066D5AE23CF81AAC322BEBFFEB9CDEC9317925AC69F04E6"
00919     "0E565EA4C20C682E33F6E6E4986FB9F1F677AF03612AA49AEA27BEFB6160A2391E44BCCB8A"
00920     "676E7DF729E4FEA84CBC1E7A24BD20200E67E22E071E4C2753E1211F50F1C79CCE26FBFD21"
00921     "E9A57F809F193586A6AEEB62EC6B0D232CAB95BAB7CD79A1F3CFBD113DCCDFAC705ED3D715"
00922     "C36A2D6DDD78710427B6A8C331FC525EDE113128AC69C538EA0D1A06844A0ADDD091BA3870"
00923     "A00D359FE00D74745BD7C11E76AB9D22002918FFD216133373C37B229BB2765EF6B67D0463"
```

```
00924     "847191D0413433D7861485BA603029983BFD17DA369B7AFBFCC64FBC04710CA71C8EFC766B"
00925     "8D4745D8F6CB78F00973A1EEE7EDACBA7324910A916CD925F3B37DD601DC6482408E964F48"
00926     "8FF08F5624D279729AC1DAB38226586F042989A480F236A874F1BA682A367F1636F877B7C3"
00927     "D6498E7FA951D8A0DFAEE1519AEAD4797D378CE77A49F020B839FF01BDEF59CDADCFC21CF8"
00928     "F678BAA531A4B2356EED839C1DC2A39EE53F19FBE4311F08A47A3091760F3E75FBC6DF3CC5"
00929     "E9C1F23F0C209B4CA563FB6EC7BEFBB6DDA80E86328613EA74DEEB92000868940ED956AB72"
00930     "69DA335D402AA54C50A0591EF0C21EDA4DAC57E547DB1245BF62069E715FE2174A9AC9D982"
00931     "379EB9D422A6C03845D14F043DC55ACF989D262D6458D07A0E198FEE889D3491DEDA7BA25B"
00932     "3C802A537327AB8C102FDA6C033AAFC6BE8602FA7C",  // PRN 25
00933     "512FA9AA09712E0725988ECB3E10F8C2A6AB952A9810C3B48A372EACF9FF0E006B8EB2A872"
00934     "1582320B3F6EBA8928477A57F3EE089F5CE05DBA6C03F3DD5217D4A6A20E3633FE4C74E75E"
00935     "C852704F0C8BC005CB79BC7D0B2EFC25740239E3F9C328BE021DD9549276B2C0C8EC3CE2D2"
00936     "8E56096C722738FD5609801B8497BA11D4CA2CFB2EC8CA16182FA4FEFCAEA1A8C0D8208FD0"
00937     "A3BDF519DDD609495ECA765F66665E6CF33BAB0F50ED40CD59B40FC120A90941AFAB8222FC"
00938     "1E541423045D1656C4F673EC9B0B0DCA87D5BD5E9C9C21AB203669751C39B831300F92AAA6"
00939     "521635BD53247B63F1B3BFBD1BCC0EF22EBDEFC5C5848CF2A73BB878DF4D247AE45B9DC651"
00940     "7B2173BFCD045A223BDACD719D55D23F6D40E1BBF4B4AE36DA16B1A180E8535544C5D29F0E"
00941     "99A17A763BA55ADE0FC297DDDB3C61D0931571F773807101FE225C6310E1D01E320E9EEBBB"
00942     "750D1FADF4E3CF0E2E19DBC5E7BF273FE72DFB86029139F2CE11082D029DDBDE15EA300D66"
00943     "12AACD70AAAF0D78E4238CEC645DBB69F7A21CDDEFDEE1D243B200ED84CD81387D7D0F19CB"
00944     "8693D452F481B68BEE7E79B6D05406B76EE5AB521D4BB1A5843D87DD789884DD150760CBAA"
00945     "5FF93BDCF78ABD053E9699BC63D8EEA113FD217BC4967AC4BCEEE02B679979AB757BAA8091"
00946     "51AD0969E7E723223F5F126BF05E3018DF9F0986F832B50197E45AFC40A927CC1AAE622649"
00947     "DB0B3BDA4492E4650FC8118B3AB22F97EEE61FB3D8D76A7C7A317D2BB18A7CB90E48204DB7"
00948     "D8F8210ACF33E28C077AF20189CD1EC85B8C1A0B3416D4FEFB6212617D2C1F7BB646A958A4"
00949     "8A27787563BE6E94A234F3CDC9906F7DBD6E9E49E3891615F5B16257DEEF7255C89A055C5F"
00950     "6598EC36872E5460C16101D39086BEBF0EC80CF8C9AF4891FC2629CD14F1A39B80DCE89C85"
00951     "ED26BF28CEE87CD914BBFD0486CBCCA132BCB4CF88253AD6D5D305425D286532166C3DE248"
00952     "8160694BCBC20EDAC104DBC8C056D6D7420A0B60E9DF250C8D07E070756767DAF7F24F67C6"
00953     "1767362AE647E4025AA34B4BE95A68BD54EDD78FB45932FD74548E837C344591C8A0399046"
00954     "D69995C9B4BB47A66C2BF8B659CC2E49595639C6AC773DF17EFE96F312F5F592E54977879C"
00955     "840B78D551C75A7899A2C93AD600AD79BB06FFA680A2980CC6640B3F6647B416B9BEC68BC1"
00956     "C6A73115484397004C98F9F0637C2D956B8194EC55F47EABCECF3DE06447FED6742F39C517"
00957     "0D0EA3BC5E849F43DF36251A75C4C4F723A994E92BEE217C526AEA701D95CF2BBD67FC4E73"
00958     "DEAA2FBCCBBCE1029078866584718431B1147DE9B39915ED65306B71E90AB557E3B33E28396C"
00959     "94992B66F5FFBDF0946FBB1956198B78013F7BCBB3F39A65DFAF3E81E7A058AA21536D45EA"
00960     "46CC12C22CE15756A6E3160CCE5A8CA2351AB2CDE145186535AE86416BF29E2FF87C5742DE"
00961     "237D3F52F1D96E10EC2E65A82F8E1A842CF99BFBAE0DEB1827243B6E008998CA082851489B"
00962     "E74B71CB59EF909F464A14C8DAFE4859D1209D81430C3AB52DA3F5124D220124CEFA0FB860"
00963     "03DFBAD6217248E456D05B7D26E27C9922A3F186137C6BA98EB6C79BB20E20B77B16E6E3AE"
00964     "1EB826E8E83323A1445DABB9616DEBCE266D142A7877EE8050C45768DBD0440EC6FB577811"
00965     "01F70C96D45D8C5001A49CA5CD852E8B20290145BE748D6244A88C2BD8938ACCE4D68D4A65"
00966     "82E0846F63613F70C4F995F673A98C0B09C29703D3A5D9FEE87AE2B3BFF7B46A674018F146"
00967     "6C665CFE0C94933EA98E8950338893E6BC9007BED4",  // PRN 26
00968     "73F36B0BB91CBBE1AF4467EACB75C940A846EEB5AA2B2BBA6DACA0DCEB0E21EB94AF736D35"
00969     "595390978563CF7400A3C87E13CE9AC82A0A92A960BF3DA6FCF59121672ADA1E834748056D"
00970     "2F490F66955B95EA45B26FBA8206A3566B44DAD1FC6B41D56FC56F6F38B8FDCE5C67646811"
00971     "47443A7268C622BDDD44CAFAC9D7BB4DD6E269D80AE94683EE1552329BE51607B9DE76B0F8"
00972     "666F88B37039FB02421413DAB547F6E959EAC432021D33248ED0ED2046694D1509E743304A"
00973     "811FDF0CB242ACC460611652784588ADF7527D0A7810482C8C5268C5DFAD4A997D74A39989"
00974     "988940361B1E928FA53E733901DEF53DD400E71D21CFA48CB470B41690EDB1E59E40CC1F85"
00975     "02C47425B17774B4A687031A839DB2FE16B6936D8104D6631E183F7AE305FF3BA5645322E5"
00976     "10E065E8B95D26224014629CFAE284A419BFF808DD20F42EAE1C4B041606CE1113365A32C4"
00977     "7DF49BB29610A1E01F818513FB4FB2F79DBCBB171FE0BFD029E3C6B08DD26BFBA0B56DE26D"
00978     "582820CC746489BFE71A9B5673787D267CB7C6AF50EBBC37A8E9C029213246D815CFE1B812"
00979     "ABEBAF66B7A6DD001765B7BE6E60B9E193FC9F9E5B605240704674DFA64C2029D0455C12A9"
00980     "D7CFA62F38F5815CCC472400EA13E077580D3EA2F6CE323684E2E8B4337265DB11EA5B30AB"
00981     "C4C192FDC7CEB5627353A1359213EFC8E9D07B92BC73D36FD2A4DE2113AB5719AA7836F665"
00982     "C44248F1A20718BFE8465FD9AC11323E090E811530F77DAA5DF187CE29C16A877A8FDA753E"
00983     "51365FD425D771C5B310ED2B5A1A27CA21B9BB9F49DFD9EDD6BE88505EB3FFC1857333FCB6"
00984     "6EC8B10E104DE9CB27BA32FE1F5B6E3BEF7DBC05605B081FA481079A2D770DE380201DE980"
00985     "C92E6C5D7FC7995418E708B427408754C549BBF351AC7E165FD2319EEAF0E0B99B5FB093C7"
00986     "751EB9C040FEBEDC45D3085A8726EFAE9375299DD657C72320A3F4E2D7972CEF3C9E8A8F00"
00987     "76286DE2401A7540833792D2AB5DDE950723CF502763CA1777AF288A157E7DE6FC701C6A26"
00988     "DB34DDEB87E1ADB340B4C08CB2F3ECF53694A8294CC90F0E152AA6F81F77E8EB0B902A3D6B"
00989     "1523972C50709E81F4418EDC6786A9D4ED29E6CEFBF236B11C026804C0503EF573EF1A26DA"
00990     "78194EBB438754B48269B18C43AB771424006438409A4C659349B405ABBBBC70ECCBE922FE"
00991     "4FF3EEA94EE9D9BE1B8753D2EDCE0E955A51EA839E370908C3D6F02DBC5FC46898F1A5415A"
00992     "1F7F7D0E2A842482E990E6BA271A47535D7D9E1CFA8E8C3741249616F321D624414EFA4F4C"
00993     "7C7EB50869722B8D55D9EB24102689C7D042B7169533827C72C7E9992CE40EC693500ED914"
00994     "965DAFF96655583FDFF3942558EBD0D4F57FB8A640C5E4C7B584B424FE46153731CDB72D05"
00995     "48F6988E84B30C5453B7F7F0AF7C344AA42C430ECB068E2F99B37736109E4D9B6182F2E0F9"
00996     "5846EC6C796F02BAA87023F5ACB5B53A6B561BFC66011A86EDFCFE457CBD06FCA17BE8F978"
00997     "F15E8549AB737D92B9A674E560B48D04CDCA58725EF2DDD7A24DD43A39D15B6A89AC8E219B"
00998     "D6753CB99631F05EB7641AE249390FC4A4E85699405F7B2874EDEEEEA3E778B2D99F9CCC5F"
00999     "B1A28042088C609C7FED35CD0896C7653E317A32D86860B2829DED99DFBBFE27A35A940552"
01000     "1DACE8422D49D2415CADB642BACAE22E5A6FB007396CB84CB1952D3A240BADEA3F9EE0A303"
01001     "C72581E31A56B93183F784FC7A097A348F06515CE89A26CAE99D21AD5C17301935564AB564"
01002     "AAF4C6290EA68626EAC0A236572DC22048DA24C264",  // PRN 27
01003     "2D531725771175DDB1061663798BD8B3A0782B5377F7488AD000993DC09ED4C9158D498E39"
01004     "1C9C750ABE7D50144BC976C5CDD80FCDF1732F3E69313BD4D381BF6AEBB5CC3537A2366FD6"
01005     "E49FB8E71C106A6585630A67C1B4F9A69D2C5C9F2B7CB3E2F3D41306461507FC35FFE83FD6"
01006     "8CACCD946D7B8BF434CE4E837BD0AB20ED05F726725AA320956C41A4864981E7866C120069"
01007     "821F90A24E902D3AA2DD3039D532DC306EC284612E9DAB3DE86C29BBF9C9218BEDCFCBB286"
01008     "5D955E0508E4A825CB00BAFBD81DB63F1105B02181A5873010F74583E4FE8548403973D9A6"
01009     "34C6C1C3D02E04A3D9D7FCCD70CDE6A129BD32D4075CCC588A2513B8E2E9458EDD68378D9F"
01010     "249FCD11D3304B75D2E1682FCBB87B218440748A8FBBDD7E9C7F92ECE1895B1DE1FD5486D9"
```

```
01011     "0442052322BA16DD2C8516AD915DDDC9196809DB47494722BCD3248865D159127D216A34A9"
01012     "72B3738231ECA3665BB683A3E5CA8287906DA73A9BEA1D01C35FC56ACBCD28689A11D10796"
01013     "B3C2C12F3571F72C5D4F7E38A9E993B675E4D11FA7DF38513ADD163F31B6CC8B9DAE7CC1CC"
01014     "9C1EFEF74366EF346F72B31875A12146447E5CD2C802666F67C8C38CAEDCE4DB2F08AD32D2"
01015     "310D5A7FCE33060639C14427B04867035DC6F735448102E2683694213C490D24AFFAF43ABD"
01016     "44046B00286BFEF4BC7EEDEED09E17EC850C5F12A1EBE4F923C9F576A77069149D45B687CD"
01017     "16E8138FD79CE0AF24ECB586FE6E2097B6768C41AD8BD39509B49AAA6FC1092786504DF3D1"
01018     "172C6C4FBE6741B0CEC95012528003299CDD539E3E5DB4A051FE2DED59FB02699098913DEC"
01019     "F8EE2DF6DD9081F9384260F21D242B92DA14F998F544835D7E79A9E7B99A859D3D6A05A55E"
01020     "FBAB356D9F8B3BE652A7D4F9E7B082DDCC769CA69A32E92E01AA2C7A5DA908381D8456FF3A"
01021     "31CAEED50ACD72477FF235EE56F8F3DB87EC793960522D8AE35B36AFA6460E82DF18DE082C"
01022     "E3705600AEECACE7EA046F99EF1328E2A15EDE3CFE3742ABFB142EFD4A02F2B9FC7F8C73C5"
01023     "E6A16C61D119C48F2DE667FFD135C0DEDF914D1096D2F0D6D571D26D95CDD13E807611C0A5"
01024     "D03C4B6E0099783568CF23DF54710281E1F688498A87F5D99F3CB7B226EB32623DD1BC60FB"
01025     "7BEDF8E9A9A48604892CA3781C4829216955A546C6E03B04AFA167F3F2ED8CC830FAA91DE2"
01026     "2A575DDCACE10ECA494C10541748362293F818AB2215B31A96181EB2251530E5FA74AB7130"
01027     "4295064C097D14CCA4267661F5D6CCD956656C73F60C022B633085571F5EDBADCDF08C30B2"
01028     "147D7A8E54B5EA85B37DC780986304E2E5ECB372BF0304E0240B20D38F69E652D5A96CE7DB"
01029     "909131B34FC9F8629169D3661950AF536DBBCFD85C1EF14AC7B19031D2A52DD288E6F559FC"
01030     "F2A99DF11C0BFBE556804FF18FFED3166BA5EED0E933FD3A3F41071AA1D62A9216851EFC54"
01031     "2D15C59CE9D7708AFC6DAE0218C08A237EFA099D08994BC0E8EF5618B7A61095D536038938"
01032     "70F3DEEACF2A5DC2DF218011F2F63BCC868B668C88402C5648A6D76A6E86D1F1D02D4F4ADD"
01033     "8E860FB73CFACF5882FBCEF80CC24A2F5129861E2E0A8B868CF8524CE8ACA65D4D04746CB9"
01034     "3E0E61C2E965C198435796B9E19CB2450717105B9265EC4E344A86EB273E03F646768E3C29"
01035     "975EF12E719C0BD7E5E024C23DF3B754E5DF6CB3F8028E1CE3B2AF006906FC074B91CB7BD6"
01036     "5C73903B922D74733AAFD65E79621D0C287FD28B8698D6EE00C05407747993781C7EBBC76F"
01037     "CE86663C608516564910AA3942B57037E89D3A6344",  // PRN 28
01038     "EC83904B22E1DB25A4BAB428325537273B908BB5F85BA90A31E93839FC2477BDFC873CAE00"
01039     "3AF61C40B3F1ACA29CA2043D4825DE28DD691645E3C322B88F5466FC52F85EB5B09B631569"
01040     "54430C2672263A37A2E1302ACCE514D29207844AB02AA23A7BAFDE44FF4D555E36948F0771"
01041     "F52F1AFA6F8BAF1D69BBE4AB1573CA09C5F4852D16280335FB4FD1D4259989E893F4A017C9"
01042     "C5013182627665D7BAE2FEDDAABAD4EDFA810D0BB1462161F01457FFB24F5E8C845237EDE0"
01043     "82F139653F0ED987F9BBA3ACB4A7424B721E8DC8B0E0C824D4477A6680C405E4F92901C272"
01044     "6C43921321208F08B29B442F01EA554B1ADFA8FA04F0EA9992739DB100DFD29D035522CD4C"
01045     "F946DE29CF88FE047F43CAD5A141705FA6DB068B0BD83F42EB7D0EF8EDE884A4975FBB2C02"
01046     "8B6E15FA4872CDDD61D5A7DBB6332638A3B28DF5F5C446074E3CD95360EC01021F8A94496A"
01047     "9FB11C77B2FC8C9D14BD67201C54CDDE046CEEF8912B6C17A0CB99C3929488C9A51D048900"
01048     "809E1B66BA2B872B6C421195BA6A4E386656DCEFDBA6F6D6E5A9C175231A03FB31EE6D2DE4"
01049     "911D6301FC11727911338896ABB2C44B3E136DD547BE37C5A367DC5A0C9D4F1482B52FFB7E"
01050     "B32D4749F8B1D5FA808636B675582B58EE12843052595788 67AC6DD343F5A58FF56632B707"
01051     "A0C0223E533C1D2F0862921A588C9A43A1EA927EE8FE1FEBC8B625EEE9165FC14E331E8876"
01052     "8F4E26BE029287C5F317860DD5F435A4757DAF83ADAAB3C14339E70F523718510F3FA220FA"
01053     "5FB884B7C87B1F24129150C0D35B3854A341C76306092CACF7606D24D419763E2BD3B6D3E6"
01054     "50DFD608F753E6EA1F7EEB8117A2EA248D8ED2EB51C356C9B871925257E64D729D519BCB67"
01055     "3BE0F1D23077BCE147BAE7BF6DC26AE81B823F34529569AC48F7F7D161E53ACB246AED762C"
01056     "2EDE7D65BF395EB4F73DA983AA5E949AD05242162F4C092B9E974BB42A8E18237AB5696AF4"
01057     "0DDB6F3C94BE8185C10B4BD1236B4366C85CF309D5B277ABFE64265C5E8E623FCE2854981E"
01058     "D73C42BF6CFDDDD7D14570B6279F776116BD3E6900C5BC507767792829389AFF4FD14C532E"
01059     "55A74A7F52004B3122155EF078CFDAFF0BBC1A141C7C78E38AA175AD27B8A0539BCF0DF0FD"
01060     "B8E3AB2308916AC045938905CA6509CACEE4F771F4F0534F409E8D07D18E1DE49102E431D3"
01061     "9369DF8AD109638D2E1DF7C00921C7EE29A26DDC1BB9C83B151DF16BEE4CE6241B23BB4B0B"
01062     "2F1AB0FDB2FB92CA538853658BACC738A594532823088550F6E2AFFD38A0612FD94DCCC8EF"
01063     "8866E6F668310996D681E466881E8F2F5E547513C45B7722732ACA968D2D9394CD22F32CE1"
01064     "E6AC0F73D91C85152D44F96DAFA3D47D7EA7C33C02F6ADDDDE8FBC66BB6FD107FF715836FF"
01065     "E03E62EE9DA9305FD1E4FA10632FDF8BC58A0ABFD11105E8C21AF37E72E7323D9912CF1398"
01066     "266E970F3AC2787998638ED86F0FB7DF8F9DC2DBFE6B6CECD3B679D8C8C2D9F4F6FEE3B91A"
01067     "0DA0729B0EB81A041FCBD5A39CA6BB6ADF8A56BE4539A56137E60E57A6D7C0F449945A48CE"
01068     "FBCA06746A39960BC24BB3DDC06DD078A24218271740D10AFA5FCDEE923706DD125CEB6F52"
01069     "A86F4D9A10B6DE70A171B1801AE719D74A31C1F3EECEA312D0436E3DA16FF4D2723A394E41"
01070     "53D08EAC9950DDBD75ABB77D181E789AF8A17A88FCCDD530E6ABE74CE034F34F33B88FBA41"
01071     "6BBDF1C54B0F9C11EDD1648478CDFEAB1CACB493A30609F6E76F7A01A857F374C66034170E"
01072     "817E6330E8EB1B36741ACAD95917486CE0887FC090",  // PRN 29
01073     "380374D28B44F31397486F759F9ED530CCD71F0590BE4721F98C894AAB422D58CACD3EB2E5"
01074     "6F1CE283C122A91A45E34CA340CCEA1D9ADCC6794CA074D5D71304DDBDCD5450D1D1EB18C4"
01075     "C8A191C07D7B823C6272955F1A4DB5681F8320E975E28FABB12F8BFC710C97D176B4B35113"
01076     "3D719BE6150F2FAEF09850CBEA485BF7E5D7F37C180DA94A9566A1B5C46098C68A08BDB1BE"
01077     "347CAAFFF3EBE50813247617 46DF895F1150DC87DDB185A4E52804ED72B0CB1C6E508509A1"
01078     "4A9E68072A419A43C3950E018644CB8FECD125053A88702A64DC7E07DAFC226C23754EBA86"
01079     "3940DD64456C6F7A504E82998B61BCE0FEF250706938EC881DFB6C6D0AAF82AC842BC717C7"
01080     "47EDB0157FCB9BAF5C2FCBE0E384A8C21D34C06061C88588DF799933521FE8B0623FFAA6AD"
01081     "72991A4E41CD1BD1216A4AE799A129DEDA8FEBA5F0E34E2C4BF880DD319EA467790810B559"
01082     "C98B44BAC195AAA07D7D7BE429EA79900B1AD6B6591CADE647F186BE41C9A0831620C63800"
01083     "C72EBB52B45C5DD6B237246CBE84754029C2E9066F62E8E814EF6A679F65597237415 39B12"
01084     "9A3BD1097C65AAA31A7C7A73B678835F7F22031A1672239EC2327464677A86EA701782910D"
01085     "A91BACDAE9952C17DF09586604A7F978A42438A3C96CE6AC3040267EBCB7C4B66DBD65FADD"
01086     "00FECC58CBE8DA7B31FF3945B442B6EDC9BE5D72C47117EB4D3774C4DCE0CB369676C4ADC9"
01087     "30028B9C5131661D826DD57CB5F77FA319905B7DDFF947201159D5952B343E6686EEFCCDD6"
01088     "C000AE170F94E78F1F0D1C5C2176F47EB8471E3D6EA4E23EE8363E3CF4F7A187D8F0085308"
01089     "29673109D8411855ED62F7D5636FF00D462E05B4834D1DEB26739B772088E183AD9A572CDD"
01090     "D5C4A9AE62397E369709299C53B3DD1A821F3A6AAA79D490306572265011 54F804306BE09A"
01091     "F3551EDF13DDAE5DBF88212F96CFB7AB31FC53B227F8DD858A434BA1EAAC9C7148F771A3AB"
01092     "6E7403FAE005FB72516E2823AF3664BC62820C2D647F91F3A3563CD4042374CC1B0753A683"
01093     "A1726CA900F9989DB589619D4E6BCBD3EDDCC5C02A8CCA0B9953917E5E30E6FA01998ACB54"
01094     "93E751736B4F61B5F5A64DC284662DA7BF3FB48AD934A42BCB005A11D81370DE158BBF771C"
01095     "8C536DC7F2983337E42694988DFB59B36923C5989D258FA9F1720B0BC6317AC9F659E3B48F"
01096     "D381323CEFD3A5633405EC1EB1B5D617C18DEA13DF44A76F07001F3CA6E64EBBE1C9B319FE"
01097     "2CFBC68DDC8A723840041EF27B235F7953BB65C6C01100BD972DA6A1BF097526C89AD4AAC1"
```

```
01098     "F74A0F2E9CCA0BCD83743A963CB6A1F493B8CD5FB468E659D25896F2759CCE33E6D1A6F2BF"
01099     "53100033986E0FB13A66A13703D7453CE3A95B584BD67113D556CF793D4BE424B3B13E608A0"
01100     "A78152C370E37F1385C737805BACB654F97DF97A4E3C1E330A864393165833C00A374B9CBB"
01101     "A6AE1EFA7473BF7FD17C95DBC6597E66CD9F5BBFFAABA7D93592A425CBA62ED5B2BAF5DFFE"
01102     "59A83F4E3E9F175F12AE98A15D522C5861E30973EB4141841E8C3F8D38A9491E27165B35AC"
01103     "87C5A82DEE88F5EC8E34105558F137954CEC51138C0670F85762BE7EB5945CC455807BBE60"
01104     "75A6EE882948D5D392281BC134A9E540FD0EF98616964F01ED2FDE29C44EC6F6BA0718AEE4"
01105     "40D6D273A0D726D9FE0A3C20E2D5109C472FE65BB155B5A69DD784D25C8CE83F8A38BCD23B"
01106     "54FC74FAAB1BAE3055E4E470514E7EC5944DD8B3F61086A30771A831524B6AA1E3871BC851"
01107     "D67179384 31F86E4DB31BC0DD663C8267835F87934",  // PRN 30
01108     "46B4DEDB85B3D8B7C21C2088EA9AFC4A4D27A43DA786B758F04AAF815305E17831DAA5C594"
01109     "E3DE143326A607599169482BAB40DB0AA54AB29536DDF5F87DE9EBA5E799433D3727B73EFC"
01110     "D1692C10C455B0403ADC46CAE3E75A47455D26CC5A1CB0A8F4B8FEF2B8675408D4AD880D4C"
01111     "F5AFEBA3E4AAD5191851F6A9DB1744D846086B6AEEA88EECBF77477DD8212C291473445371"
01112     "A2F1471BBDB393BA4CA488DED41DCECED130987D7AE96627C1A7896CA61BAFC0CF51C9BBA3"
01113     "2AFBF2F4382F64C271EE9EA3A44FF8FDC6D503EFA29BF687E26ED886BA0669F57C41E15C51"
01114     "0532AFF0F07482356F0A77AECA610251C0F5875CAB605D7BBBC4BDA946E1F15A737CE38D8A"
01115     "E4BD4943E17253BC8D82C7430DCCC05B73D011E97D37805946D54C13C53896FBB9A0669C51"
01116     "5EEF13D0F79511E3B1F358838731D98C530CC5CA6C20B39E0E284573CCFB106426CF5EDCFF"
01117     "05BB0F877FA465FE2170C3B8E9F3CDAD14BA1429A916B4F69BEB46C4550860B4525070282A"
01118     "4C6911A2BB3545ABBC7BCEFE4D139AEF824564023A98A601437AD10DD8DF19577B1F9047C0"
01119     "8BA199DA243DE8B133325BF5C39FFB80AA6444C916BAF27E9E20EC0B611819B47B1AB0DC85"
01120     "79A7F2E034D90E54CCC2A551C02616E3AFD57B78D3718D7A29C0BB17D2AA14A1429B3B33E1"
01121     "C3AD7F0939BCE6969201C107086AECEEDE59AACC515F4CD7F80419CEEB005A85E6E4C4DBA9"
01122     "EE594E28FC6FB5717A59098AB431DCE9B0FDA2ABB0190412BAF0A8DCD77E989B66DBAEB824"
01123     "78FDFF037A05FF8E918EEF227F6D4966D0897053AE6537C98EC2DA1E9C642F724B500D8D01"
01124     "9C512C0B4EB505761BC2A07BE86A9A55DC5DE683F6EDD594C73BAB7EDD9E4556088C2178A3"
01125     "F8CFC24CDBCF2D80B6F3DCB43280A5D0977BF9EB98F31DB9331445259007 70DCF8AD3CCEE0"
01126     "AFB4673F644F733A1D8E3A400F7C42B860A0069AE8CAB16D0EB598E692DD48232DB5E37BBC"
01127     "C4D30C19E9303C36D8D9A63AFD2AD6E9026D8796BF190C576CA311C53FFAAA55EF42E0C3D1"
01128     "83805E451924E56F88F561A30E0C4725BB1AEC9BCB283BBED3CE5CF14680495E160D6DA33B"
01129     "6C84501EB9194B5D12F0625A6B9116B3C04EBDC97EAE7659F1B456DE06BE24ED6831A2B2CB"
01130     "81F1613C8C43A9DBD67BAC6214E3DDBCBBFA8E30126F4F84E539DA744B49AFFCA15C19AD66"
01131     "0E05C19FA00AC03BF6E69CDBB149A78153403D9BFC1976242A0E04D6BE04919D460111E974"
01132     "C4B291B666F41713923194E643E2F60EFA566D4924D0398B4D6EC9EA8D275D0AAB536898CB"
01133     "CF3A877ED1CDBA31B552086FE54958549C86D5A8AB6CC5C342EA04D363A85AE55EC0365A7D"
01134     "4961580AA3CA7524043B0895B58F6FD508EC9D22CFE664073C71FA9539405C282ACE1E7BCC"
01135     "F1B49A17E0B7CC9F6651D32D527F50BD85680718ADE2DA27D46647111DC540FE2C8033146E"
01136     "6AF88533B4C17A5E4D5949C4762BCBB3813899FD1B63668943A57AF41F5A4A8302F370659F"
01137     "1CB02B009FE83B334B0B4996F0932E5ED48D84E28FC0001E015FEDC43272683323EDA06F67"
01138     "41C20381E3C9A8B2A45976525D5C2EF680206BB8A0FA682E2832D7695883F31352C2660737"
01139     "B71D0EDEDBC1F8FCE99649AE962A7F7E4E88FA01B0125FF307A20D8DED9A461223AEBD86D2"
01140     "7324092E0D1CC4CE094DA8908A30FEB78F4A5728E633EF43AB9D86CEB9D902C1494A6FD783"
01141     "8DBD3C64A3F29A051251344CFE2D28A8CC0ABA9219258C46663B203CFFD7883C058846705A"
01142     "16E2DC4D95B7275E56B518AE13ABE5B7B2B5ADC1B8",  // PRN 31
01143     "6C01D916F94AE76996498DC689DC21382F033B000C775A82FE7555C55A4336EA42BC13E11E"
01144     "1E57618CCEEA206A8A03A190FF4706C676502A7FB1EE409380702E302A93DADCB2B6563EEC"
01145     "614309567A872EBB3267027D140ECFE30F295AF75433FAC6D8B897B255FAEA971AE7F69166"
01146     "B9A9B8ECC4CE13E8227A2B0C48E0418C5EF5947F8F1B077F2132FEF89B41F702D966FD8132"
01147     "483B241FA069648428D358C31B2AD22EF4D2C7AA84AAE5AD4F3E2A299F6FB52B350000DFF0"
01148     "EABAD65AD5FAFAE468970254F98B73F24C9CDBB5588D8F4385A0EE78E935C95EA35C76FBDE"
01149     "17621116CEF94451645AA57811D06E269569419418276EC6E3C5699CF7AD4336EF26A791848"
01150     "7E3D4BFBE522D9FAC5C1DE68E1B92DC3BD477EB78F3B2C27175EE8E89063491F2F99DA575A"
01151     "F8AC42DBA237CF991413BF5029F81C4E9E8C4D6320B931BB6316D2E593972F110386701782"
01152     "C32605499EABE557E86810B7B13BCECC9393F17933E0D690C47E782F439815678AF96BB122"
01153     "784CE63D4E4F541A74ED9E026B8582331D044F3CCEAE05B2EB891B02EC27FF0A56252BDB16"
01154     "B6315858D3D10D1DD852AB5E4B82E6366DBF1C14D5F9F10549326E345C1DE94D83851D3607"
01155     "3A1A3809218C1DAB9183AF0F605A382DA21B4766910004C8D7D1481FEC6FF27C9EF80B8E4D"
01156     "FA4C5499CC9E67B4A8E2CD42C78F71D5C0C6CA873BE1F709EE911C1592BDCD8C68E2B6F4DD"
01157     "C012C2B7EA4401940433B24162865D60FB022D6B617CD95108AA582ABFF8D64C7129992435"
01158     "049481AE803796161DD928E9979AAF636D2039BA24E14E7B89D2B7257BFAD10AF8B5AEF678"
01159     "8FC731A7D9553AC47F72319F98728B6B758BDE633FCE36F4B2117249F5DE167A8814381E1F"
01160     "3C547DECB751C9A83E711068B801AC6A317F1840F95FB4196B2AD7C3C7CA5B254D62205245"
01161     "82F25C6CAB9E4E77CDCE15EC40FDB77350362AE60145451C1DA6041B2F088105BEDD3745E9"
01162     "637D7E1D1D5B7F1E234BBC1CA693E28B455246E2D972391F496ACE57C91D4D5B5049FD41B6"
01163     "2E165CC9BD02A1F68F515DB7AC6C8CBE5C2A13F75A5C0AD56C9731F191719AC47FBE61B1F4"
01164     "030823C92E1A254A22F60F552C7926143C4C3B8D49A8C647748426D3D7A209696293735877"
01165     "822B2723CFD0937F6A9E5789FE023EFDAE841D8A478A215E74D74A7D185E1F0B95D87F3F52"
01166     "D813D11B2A59485A6B9F42EC44183675937209E3EB1DD03FEB96A3013CC7DD0E24353EFEFA"
01167     "73E19926A206E85E6311EF00811EA6F83CD137BCA2B01ED116E9BDEF46EECAB8E463BC741F"
01168     "B3AF8845B93204211A581EEF20B0A5BB39CD436786201E81CF1220BFBF8CDAF8C6F7A0AB10"
01169     "6236ECD19DC74219844AA0471FD411A093148DEE650C3B554363532A414D08B55660212F89"
01170     "ECD0DC1783173040F1D73C6B484CCB2635587265745FE32F4E42A4AC792D4189AEED7E932E"
01171     "CF45827B49696FB5EF98135250B7337E5C50ABE2E3316A8B3CE0D3C1761FD7161D7832BF67"
01172     "B2CDBFED17968679534797A475CA1EBE982F36A8CF1C4DDEF15A64CD0D553DF8BD7A98311F"
01173     "5C26B4D80B482E6942358B16890B3BBCB3ECF08281C7DCFFF62EA27DC1D357A55728CFC4CF"
01174     "75507F0FF9AC1B7ABD11653BCE119D90B8515D6883EF14B749547A2B908B2ACB7726010A69"
01175     "755356528243C4C199544BAD1F8174E177236B45B4949C206DB1AD27BC3C3B1B52B5B717D8"
01176     "F928B6DE9A2711E329ACFA89E1F16AF825EC0E0E97888F759190753020B90F4D704B2EF364"
01177     "C9543AF699206960F0F339ADE71DB75D91671DCAFC",  // PRN 32
01178     "0E0BB666FB0BE2A72FD69E9D35984BDE044EDFE05515CA4CEC9B5A1D6319DC4C115D3291FA"
01179     "7EFF5A972182CA14B828F9B43D9A3664FA5A0D2F0A7876BF5298B5E6CA12D98605F09E8CAC"
01180     "C18E06BF93A3ABD6A2F18000E9A3AA000CFC8CF1F49D55D3EE0B943083765B717CFAA30718"
01181     "B5C28744F045E563F591A8AD463CEA6D0305BF06CB8D90C2FE1BC929DA8CE85B1A673E5270"
01182     "4042515949BB706E68A11417C9BEC9370DBF697C615BEE56ECF4721DED98A0CF07D8B97443"
01183     "C7C84D7FD4F4884E9DE863ED9009AF9EEC3894B12C1795E05F48F02F589EFFF280ED660C44"
01184     "9210A1B38AFF929F1514567FBF0C6BC6DCBBA3AAD6263128FE04C82D9EC2FB8BB240A55F68"
```

```
01185     "9F7AEBEBFB09E89EEA815055A3F9740F2C9D4EC812F3D33706E5C06F82B676A83B21EAA7A6"
01186     "703B4E25ACE4F461AADE0198E738739A7DC828151217E8CE8BFE14D3E8454D9B9DAFF977FD"
01187     "09EE7048C7D9C5B3DFC0A1BD169A2CD184C527E0422434DFA3AA1EEA77C4123FC77B5B1C3E"
01188     "6B888B1369F32E30F54A25B04DF5BB4277E714798B5AD040FD755E340824F7CD925DC5DE2E"
01189     "9E299BE7A13C48C28DE5B5066A3033A7DBBF86B5B715BFD8BED11B8154408C5781CEF8E9FC"
01190     "8C1042B901494A576A04750BBFB13E0DFDEDD9715807F7F44962601D7FEBE7FE7CD4CCA6F4"
01191     "A8FEE7D88EE07E1FB5CEABFB731C6F6120C8A89DE4BBD00E558D3B71895D357FF5ED081D73"
01192     "54379B290B1C81B7DC63A846596B238FC198D1E916616576C2B0ACF0DDF7C934D27C107A0A"
01193     "A9D002F91F6CED138E802CA9ADF6C266CAD94B34C1C2C45B1165D08D9DF0FCBA5261B1C38B"
01194     "6B3848F68FE003496407F8170A0E7DC4B23CB65BC02726D1A94F3B846F0684517E31EEE1BE"
01195     "0CC8B1811CF724F821062324D2620676FFAAB7B69E49E32D414D51D5804E5B678DCAB548BE"
01196     "C889B04E468C91A86B9884C5910D5B4BED1F925600E046AA6B5C7B2206F2F1AC8EC47A7968"
01197     "F066BAF5B22DE951FF5EEC7F516FB70D8B74089BCF942C12CA0661639FFA0454FB607151C1"
01198     "45CAA5586242D6CA5322DD202513D347522439CD3CD11E30BCFC9E18D6540B8B7EBAF30A43"
01199     "5570BBACC869F83D30DA317C0B184DFC65AB2A1271733F0BBA0689AFA1FC97552BD188C08D"
01200     "B532B70E8C9BC811AF2F8ADF8C7A4A59F4404B940F1BB9AEA3DF987942344801FFD875DF43"
01201     "9A3AC83A263DE7EBBC2921D619A54DBDA1C3F0A5C58E3D6A84E93159DA34F120BD387AD3AC"
01202     "F89873B12199A6E54406E436833C03CA5AD9A20858EE47ACA614697A6255E74FFE17974F9B"
01203     "8DAD2F61DA7E39423A5598F38725E7D0E044DA19D2DBB2DB8EC56FDD1BE2BED816607ED4C5"
01204     "8A702BF3C484B283DC29D23A1B78EA70499A434DAA240786A0FA279EABA37BEDA023272456"
01205     "5F466FBA00D59E6085A1E774252FBC2B39559D393E571FDB5981434E079AEB80A01A55377B"
01206     "9B0A7F0E4D4015341F232B6568474A779A76601AA4FD4A702B7BA4B0FADD21A6FAFC9CFCB3"
01207     "38C937069BA79146C5337BDE084B331462C04EF29B0B2038070642F5149D80280454AA4D47"
01208     "F2F52121A1A701339C33AE39773438F92499FE4BEA04055DEF15FE22090D27E0A68D357327"
01209     "BCF61D3EA6B334D9628973CD6B363BBAA9B79574C25BFB2932AD98E40606C72F397E77B836"
01210     "F5AD9FC040045501F910B10A4762F7C5F8AEEA911FF040816CFD1F25DC59A1E840DB49966A"
01211     "82B814FC911463C92D8A1011F277C77E974A8F6E6DC3162AA53F93E71CCB7DD9C553C3BC5B"
01212     "8DA10DB7999ED85EF724156FBFAC44FBAEB6D986B0",  // PRN 33
01213     "2708C751F8CBF4830B2A03CA1BE2173F7DAEA8728872684AF7F4C3AE960A8D104475FDA396"
01214     "967BAC4466FABBC0EA4DBB26E58D9CC3F5B9041ACD3CFF08DC81E7FA7847B9698E4B45EA31"
01215     "6541F392829872083B05C5B38CF188F23E10952C896F68AFEE35D4570F800DB755A142A3E9"
01216     "00881A897F5D55CC51863C4A2E53150E02686D50E1657794D18326DEE198972A132499C8A2"
01217     "3278E5558665DD3A5BDCAE0A0C69C495014417A7FBB7DC0BD40EEB05699772164C70363E83"
01218     "EADC06AEAB79F09BA8BF661789E0721A1F9BDB68494F57032F8DF0D329086D3F5765455195"
01219     "D93DAFA860E7704033B39EDD4F86838F8B8019D4B30DE31D7410F0EC37B55266BACF419E51"
01220     "A9A0A884B03BC6B6C6E299E991F34C7FA4CD03CFA30FE511341C19907B16799623D6F5A385"
01221     "EAABFCB78DF2C0F57C6610084FFA69EB26FB0D3F9C204A9F901B9C28D45E7FE2B5098BF688"
01222     "33876D45C1343BADBCD17C4565F67AE44C11B7B4F1A8D74947071739912D6E2A3919B8C2FA"
01223     "7F566B17EBF8B343CB6C92C8C5207714EC2ED4CB4E7B4C65216AEE46BDF1BB6A501F2416EF"
01224     "CE5FAC3970E8B4A6DB55FC83D05176BFCFF127C29B79E67D6FE6B640564639390C5F9C68BD"
01225     "160F916B4843917647008EF0E57AF80A5641D4300559DD992F6565B066882B32467FE05AEC"
01226     "7E81B7B1990D69B062C9E9E6CA4BADDAC09E5A14C994ABB007E299F2136386F9F66AAE9963"
01227     "A6DD16FAAF35553738FA01E3715D107D1AC57C22F47692315600C41533DC0151630F471C50"
01228     "4CE648FB0389B8F970435E7AA84E1B27824743DDF5F783EEC1597FEECB9DA4FD30405EF681"
01229     "8E59027E824D9FABF9E1F74AB205BD57174960F3619DAB550DECFE4C42974ED12B8960A07C"
01230     "838891BD4A95ECA6A01A59A052D19F554BCA488D0FB7C30E371C5FF02A31354EFDAB54B1AF"
01231     "E6904C53E0FB60D5F0E5155811D2F28D65D57E335D52AD7BBAD017D5A237E490DDD6BA86FD"
01232     "608CB847566C213EAE3CF93012AB8D2326D93A784650B2DD62CBEB0B27F4B4CFFF012B69E6"
01233     "CD10F9171AB683C91C55B96CC56131929492B346D57AFA1D4476F3D0D04584038F66F97A13"
01234     "64F7C54912E246965EE6813F6EB36E90649E68D7A9EE84E47CEDE3BF1B36A351BC1A3737E1"
01235     "A1C7D74F063431E6A09B83605A760AE6197C640600FDC093C58F9D5E4A1F4DC5A65624ADE6"
01236     "FF5C4ABEDA0EB1D1E4E452BF3F98247DBD07C51B71C0321CBCA7F96E1BC5161AA245A98E84"
01237     "B22B4E739099E155E432EA4159C1DDBF6F20A6E66396083893FDDDBE7C6B160955052B30C1"
01238     "04AB805F6C883264885D18875FFFEC9F12A1C7D395CFF3EAE00A3E110FB9DF8DBE5E15330C"
01239     "7482895EAEF7A8B537BAC31EE295B5FF274A32B278044DB01C69BCAB1A872C01EB8ED86637"
01240     "66DEB4ED4D0F3E5E6BC77FA0E27B85B6920A71F712F9778F968D6FC10F9A6E01F13060C1EF"
01241     "6CD23AD468FFAA78B20178C6899F3B0DC9070019EEFA269DDF85D6E30E3C73C6C866E5FF95"
01242     "0F40E658768B0EFEEF93BC31F0E03E882E5CBCDF57213B3EDE582A58DA0F9F7E574C8AA390"
01243     "4B06029F2149788B7B32812B898BCAE538CD8A86F02DCEFD384A2CAC422FB9E2AFA8421D8B"
01244     "4DC6D6F24109DAFD3AF52183A9E1EB096794AB0575A962B437F36008BB3ABFBDBC89464C58"
01245     "C68DC546D6B2888A9AAA1DEEB78BBF707002D6003CC1A12F7D59179108F7B53C40CE89B4EE"
01246     "D08A56E4584B0066414C64AA8880571BF4B803334A19BCF45A1B222AFA9320EBDFDB99114D"
01247     "7E7422B0A703AE10D932796BFAE62037BF9DB0720C",  // PRN 34
01248     "265B55605ED545130DE90D99D14322D3B9621758F5A4456A1C5372FCA1389D780F9AC24962"
01249     "B5183D68C702F6ECDBF485A4B3F074AEB2DE55FD592296F9103BBD8F0F7E1512AFC84005E8"
01250     "AE18255363CF623200684A053D4A0E7438E73B098D1112A40FC62568545509302E34908943"
01251     "2776A00B6BE015340C1A27D6566F91ED110D9173077 50A1F1C389E1880E3CF59D173B822E7"
01252     "303C90939EEE42F161AA9334CCFF11E876F83A8DF0012F70FA1DF7098DC99A4ADB690812D4"
01253     "8316FA7BDCEE5DF4ABADF1E89661D0D1A51CBFDDA8A3860FB223DF3746304FC785981EA5F1"
01254     "0A0CB707EF70A1AFFFB8E58FDC55968778E417F4697A2C2088E197277143A04196F418B191"
01255     "D6C461E593F32686E49FCA45E032DE38620DE99A1161B591924518A049DB001F14A9B57AB8"
01256     "22FAEDB2144BD5F76F135C018CACAAB275BBBD42B9ECE505C7E8F7B6ED6735B8397F2C692A"
01257     "491CCDC59E6CA4AE57471D693D0FA9FBA4F39193C853ACE2D54A31E83951C56565FDB12A2C"
01258     "F9079C33DFB40431A3A81CB747C8BBC286BE9F58F15EA1AAEF77905086F574317C1A3D5EA3"
01259     "476DA37108C1F0674C6232E22E885457AD2659704178499EC8D265264E63A7E614B83346BB"
01260     "CA21F167925D1EA6264F76F5E9F29607367CDE50E5427FA1746D70F5C55D5335F2F3DEDDCA"
01261     "0BC8CE7A6EDABC73A44BC31BED3F06101D2B79C9E9860D9F660CF0FADE47EF55DA4BA78DC4"
01262     "B61E2C09135ABF941B13818719AD3E7456F2FF13BBD997B7CF26007D6DF31C3FC51E6695B5"
01263     "B27DAABBF4443D01D1AED3741DA517E7CCF3F4D31DFC147DEE3DE18F62440C041A5E28277E"
01264     "F2785C6AAF4C1458023EBDDD9F9BE7CEA1CC7B032944131C0DD7CE236F75948784E0D6DC66"
01265     "EDF3915E1B30B8105CF16E507348CA81F56EB6A85CC933149E26A6AC5499BF7435275476EE"
01266     "1F2F635E3E7754B2C7D27538FB2941F717C44A7A9A2864ABB83F130F393965DBA7D82B7DAD"
01267     "4B5BFDC5E3406C075FA743E076947D873EDDC4F3143A605659E1D41DA24145930428F0791B"
01268     "B784C98EA6D3897B7D952029A1F5B02A0F13534DC7688453EFE1CEFEA0F9743846084AAB10"
01269     "69400990E86BEE387472E6A658C72CEB0D3DD4EF43C7B6453581E86A48B152D526F995733A"
01270     "66CB7E415F1E819165466ED314A46D1C8FF4F9EAEFB13AB11C3D344F33063BC948E8DDB358"
01271     "FF436A8D1C759AB2F049A26D474281335BC04943261239D6CF13CE91AA45A0F38950DDFD2F"
```

```
01272     "A5E739880D13DEED6093398C30F8C0E5694749B0C98FB6E59D1F3638AD6D6BB638C85D29EF"
01273     "98D6CCD32E8FDB082D895F223FE966BAFA1E016FFF5558681D06CAFFCDC00E2A52ECB9527D"
01274     "CCE00920963CE1D3DC7ED2B00A3023782ABF3EC317971E544D11F9DF3E2CB1DCD4BC4D8F2D"
01275     "A3D4A88D99B4EAA8EF808DC07EE0E89725AFE8E006EF466B48946ADD37DE462772E89EDAFC"
01276     "75AED62B942FADE7BE7AF7449852735480178B03A4917A9517D1B7EE3691EA545D21FE2EBD"
01277     "82B22225E88697C27BC085714A7D0A06A8DFCA205810B3D022619C73C0EBF5367D154CD25B"
01278     "39508676D23DC6818403F0C4EB81A5953A8C8F4849B2DDA6840E464A69625D02F6CAD2D399"
01279     "A3C90ECB1BE37CAEDBD45167C949D01288FC417B47823A90F52E50AA373098ED071F70608A"
01280     "2E1613A657B1D23DDC62AFAE1746B0999926536DAC57235280F472720B22633AE06026A1D8"
01281     "6389D432D4FAE6005C19F2D484EB68B3C033F5B89DF23F5554C789CBA7630B929B4D0EAC92"
01282     "5743F3F16A1CF7D3D8F1AA29F29F5571564FA4984C",  // PRN 35
01283     "A68E1CB79436B0575C3F49F9D7792D1E401A79AC94077CD3A75075AB7CA34EE886388C72B4"
01284     "FB982A10E1D62E98D1AF55D03C2AF5F008F240543B4B5975CDED3EAEF0FB35D2883C1A96BE"
01285     "3A41FC029B7B4F9D8A8F934BE5CDE5A09963F6EAA54481AA0B0C0E4CF058CDEC2134C819D0"
01286     "D503FCB2829390D38DC46CA3C8E78980AF6B5EA74A307B4BBF0457AB22F01414CC927F73AE"
01287     "A536DBF3C044DECEED8E4429DED9CFD013B04844A582915E15706D7F1EF64EFE9F866F68B3"
01288     "5200037EE9465953F1BCA841F5E9420D15C1D21EE71F89EAADDA54CF1931A38F1B9FADC1B61"
01289     "545F2AA6508D2DAA48F1BD247FC8CE391B5BCB183887E805867FFC6EB9A97072A98EADE15C"
01290     "202C16F2CFC5AEB50964A42E50FBAADB88698B8A2E2AD0BF307833A0FD8A2F5AEA9C562E6D"
01291     "7EED05D5B33DC639E22C3994EFA7277161B84C84E3A7D25BE76580F3C1D30DD7911CA715B0"
01292     "D63E4607B5CD81F2B9AEE4FD92477FD3F410973616F418C0EAD9350CE238C8D84935098A9D"
01293     "76216CFBF755C610E29C0270B710230961D960BE716259C9330B80DEECB7BE79F842EF0DF9"
01294     "A72BE1D200A8350B80F698DAF00110383B05B5802DB1AD3C511860F2A7C7D3D20F0FCB3F37"
01295     "8D5610AA2305C985740E243BDA7CCD95825B91CB40B22858FFE20BE59626D327FADDAB1132"
01296     "91152C5939EBE61C8349080F6B4F916063EAB31EBCB492966D8BD3FF9F9F302E9D18561C9E"
01297     "77383AE0964EDFC521965D36A8CBA9308845FB1BA10C2D818AB4057B058C5376E1350A1101"
01298     "A19247D0DB5C23AF619BD601650CA27FD0E540B692CF2497428E11EC3BB94DA045EA769822"
01299     "EF702A8FF046FBD959F4996922751E871F46756BD38A0B12134DDE60A4DFA3C9144784DFF4"
01300     "8B5505FEC3FF737D2A4641D82C95D4D334CD2110F3E8A5D1537B3FC7C040977C46B203DD15"
01301     "12E42F48DD125190BFF1D1E82E3796B7E579C7232F4979233F83C724ACBC4FFB916E58F7DA"
01302     "D3668B25CC42229288FDE1669CA26F15B1CAE44553106100BBD8D3179F41C5625DA3D1031F"
01303     "C66EDB8C68B707AAFE69CE6E59753DF34B2BC5E43AD0219268662ABA189E844EDFD78F5D7D"
01304     "9F067F9396CCF41B5FC373D093686BAD3C8E0A7C579347097E82163C0B19581EB7FEB4CE7F"
01305     "EE9E434E4FF63B27D18DE0F406DF53BFB9DD5B0AD57356A09955D4D1106F4AF4AB3989BE65"
01306     "22D2590CD7FECDA82A638DCBE8D236D17277695B602C1A11BF3A0FD16C585B64E37E275097"
01307     "5726341EA19E6EE9B5982EEE46177FFEC4EF0B5BE15FA4BAB3E0EB88FEB8EAB01277BAF5D1"
01308     "48F0338B1F8D24BCDD37171DA60D9323BA0A9D4F5342490E2443006A585A0439A221E1BB1E"
01309     "F92DADB8639BE1A0CCA82861320DFB99119FE1C4D00BF629BA5C8D7E14E046D610BD534DAD"
01310     "73ED77E66ECD3E7F0B93352D9FDEB840837CF898FD2A5CC7026597B713C4FADDA930B24034"
01311     "57AF0548FC02922C2A0DDE58C956B6EFBD50BB52C9142C35EE7E28C9CD5B60E1F4ED96F513"
01312     "CEF6B1C25DF8AA793A077A39CB55D0F4E6B31CE4C94F02FE959C95F72688986B4A7FCC29E2"
01313     "A333D56B82DBB02CFBB0C8765220F6A90A3EB3684BA4D8E22CD99C19B6034A6A60342FE12D"
01314     "2A87814C7A2695E31E545C4FD4FA67C171350EA8C2CBBA5A9CCA5C32E917DD96B3AD0E8020"
01315     "AD09AC44FB87A30FAFAA189EBF2AABFCE084E457E42A7A1EE077C92CECEEEDC0E3D9C04244"
01316     "67359FC82B3296710DCAC6BC23C5EBEEBF79CC94178A3771C11A337863A024A105A6B63DF3"
01317     "A60AC8FA355E38EB54086CF638F921642DAC166CBC",  // PRN 36
01318     "C3916EF4D0FDC771D077AE01F9F5E078E91EC0976659A02F468794C8CCEFBB5A466918F67C"
01319     "68B8DE9763426281A2D3B3CFD1B8DB762585AB799B51A74FBEA008CF72A2D8230646E876D3"
01320     "F8A7FCCF3A93B29B9E70131B2D4B05769AB350E4A4FEA22FBB8D5A16CD0F4D9769F7206BB7"
01321     "8194600DFFEECCEFE5E6C371BFAD352D743963754B1A19BDA1E2C727F0B68ED60E82EE39DF"
01322     "A90574C806BA74FCD316426F51DB2F44D0C87B505F72C937871C2BFAFC079814FE9477F6EC"
01323     "BBBE13E1E8BDCAB5051254F40024ECD5B7565100F141E7982BECAAEB0E80B763181DB10496"
01324     "964ACBE480959295F5051456A62D67A2730A741CB51C6C9585CCFA9C6784D5185F15CA20FA"
01325     "C79C18C7D590E5DAC7D7DB2C496EF790622669DF19C5D115A3D7A66717028CF6625C5A5A72"
01326     "E51FF5085BFE804A915A3F9A401C10D640AB0818940B4F5BC995CD5A713B196031F5B8A0B2"
01327     "6B4A1E2CE709526DA01549FBD1B87592745F7F83305930735 87AC464B967DD17FBA8B8854A"
01328     "DF626560310 56D36FE959626FE583880D9C7600F5045BD238A785239E778BDC56EA9D21DC9"
01329     "7C572C4DA832BFE929B0E1499745378EA13E760E8F228728C7D8DDA79FB8CB9BA135EF728B"
01330     "FD9C804FB1AECEF929D0FF0F375AB23CEAD24F05B47D8A7564C89AF207B1B3E911323AFAD2"
01331     "90E72F655E59502DC9960528E6E5C58E0F92769BC1942E3156C8AC0487945EBC2C4F11262F"
01332     "0BFF45594EED244D6B8B61E03B5A7C90EE3B0E710097C051649ABD1615EB88803DE26B83E8"
01333     "66468F14FB22B5653CDFAC57646400215B192B703B659CB892415F001C2457DE8EC0B08F39"
01334     "69582D6C7CF26DB7E5B58387A82C76F7C016B27D2ABF355704C98922EE667E1AE76EDA4E58"
01335     "DBF4C3F388423C156E3967D1E52306C84E94B061B28BBB9CB2EB6E04440215147B3DE26C24"
01336     "852AE0B710122860784929C932E313237471768789FCB9AFBAC5D7F72665D6D2C51FE5E552"
01337     "0C4ACCFC2C9BB15BCDCE756D357127FDD9FD8BB5A008EFEF7843B1A6310761E0F83FB61FA8"
01338     "E30695135FFCE7050D7ED09879D3CAF36F3B990FA3211429AC2084D8AC40CA4B49FE83BC53"
01339     "7FFBFC06386691254DD47D9699EBDA13CA00AD9F7526C54553511C14CBBB2D150E9DD60351"
01340     "67232D465FC39053BD06C4FE0A0FAD212B950DD3DC1B717F43838AE12D322A9EA1D378ED0E"
01341     "12EA74641CE0ABC9FF3AEA2B2E61CC523AC801E61E94200152F0B1C4B35AECF0AA175C8D65"
01342     "C45B2002706BF2D9A9A51ACEC1802B9059CE45FFE67B69F79AA15A17686207DE4D67054FED"
01343     "ED7374A46ED3679156ADAC2F71242173AAEDB7E004B45298BC6EC6A0521EF47EEB804B86B0"
01344     "78C8D6A2D0F1239BDDA751C96526F1CD51F28D23C4E4C653579CD37A89CBE86355B0872188"
01345     "56F359A702436A7DA65747238BA4B94161CFEEDA0922BF1326600B7E0E67FD2475BF43412F"
01346     "E5277465AFF9E367A902D1904026790F506304D925CFC0B90278D587F91D3782DE0001AB9D"
01347     "4295D67F7BD0039BA2917DC567D280E1719112A7B3C41254EF82B42653E5F1FC5631E8DF9F"
01348     "E55F862E42AE34F4D8BA19B76E66244B71F911C051DF63C04BDA0B0F1836FE3E3F7970C042"
01349     "EED4EF0C4AAAE9A490B40FCD55765C365FE1535D99BA5FD887E70529891E77809D69E35F5E"
01350     "9A52FCC12DD6032B11F7E725DF8D00FEECD19B985C57DA531EC7AB2913870CBC73A6F38BAF"
01351     "55B5B6F06A6AB853C1B84D2908EEDFFA98985625C54C5CEA882BD8366346050CF004C64642"
01352     "647C89443AD1013BAFE6F61F7AACB88A3824145138",  // PRN 37
01353     "BDC595588ABC103EB2D3C065D92577A696CB28D1A17BF96CB61DCCABBE92BFF6BDB9C278B2"
01354     "CB77F6ABB0A1FBF516395DBC894E98702E5A18EF28FC248D10807CB9A9BDB540714BE47B3C"
01355     "A4EDF433CE84717C7C0EA4B12445F25D3EE604AB598BA606C10CAAADF538756710FEE64D7C"
01356     "79D3AC2D04F95798946A01870F4AB18337EDE971C3F1958CB6DD51531CCCF357232814D6A7"
01357     "F35163A5E04C28DAEB0F8AD281620176AC90D583CA8B63D8A846F5FCD928662BFDD1E083E2"
01358     "72D5D0384AA4F1E95ADA8A8445D5492F00BE98A69370B887044F757318761244BE37A0E25E"
```

```
01359     "8A0B7F0B1EEE6B773A9399BB42633FD4C155F4E27C8EAED2140C9AD89D238DFBDC3145AD7B"
01360     "13377C1B7491A968C5B7BEB6BA7C996EE394124FBBE7DA26D1FDA241FD9A9F70CFDAE8FCF3"
01361     "85E5BE9104CA2FAC46664DE8D4A9083B0F944CEDA34A4FE578ABCA8D5DF941D1C253877176"
01362     "27046DDA105E76F7EDB619028726CC5A678ED28F93B7781BC11420B5307B22122A3114506A"
01363     "48451B2BEA9AF10C02B9E02F60EE608AEB57F91D345B2DD3D8F1D11788E47E38DCEC02D867"
01364     "8132984EF7F5F1808F7CD102AAFB0641FF627BB6D3E29E9CFAC77D650048C182B1E9F5B2F1"
01365     "E95D4078F5DE78E9035BB8A5C755DB2497DC03EF268E1D9ACB631D7F03E449A7F865AF41B8"
01366     "D75D6F40120793F11B879F6035BD9DCC1DBB38BA47E1C949891DEA435494966DCC1303BE40"
01367     "0D9DDA0745E01600E35BDDBE018730B2AD28C58C375341BEFF914C55CEF195EBEBFBEB1E81"
01368     "085B5F03D404EBAC001CFFE43596541DB6BAD58879D9B4E29156B8F7A2B17985B27583E233"
01369     "5EB8BCDF0C606DC1856BD8421C812824ACEE86A1DCBEF3F44E60E4DB38099E59D5C5733311"
01370     "27BE650E68776E609C819A80F15697D04F1A3392CCD5A021AA83A06A2AC2D4C18C5B168806"
01371     "EDB96E4283B22DBE71FC1FD43075E73E5787D6D02777BCC5E25FA4141110761ABAFEADD55B"
01372     "F15C6829898DC70FB3E37E06D7C8D1AE6FDC71408869DC97B256D565B035EB29D13CC728AB"
01373     "2310B137A361B02B772231E199A4582E08999F14CCB8A7DCBF71D88D848FF827A17A756F14"
01374     "078A28545D2403FD5A7EDA997AE2EFD48E891A116C74D9F70631F39F70604B88AA52232E40"
01375     "5CFAF3DFAB862336EF2436CCE616705CD9A871D3E25B16F3B5D54CC876D38E6F1499E52081"
01376     "0A5ADBC00B744CA190C5892A43B39A78D22F57C9867BD73A468FC0CDF0FD7622D82DE881AA"
01377     "1AF674310DD1EF1C7EA8B8C23AFEF00AB5FC18C71CE2717395738 35CC0ED4D1E60734E9E92"
01378     "600BA19139D69D88BF7028DCD26BF816F22F5490C1B4AEE92AFC0B77DA3DA9AFA0AD459D5A"
01379     "602A09847739816B574F189E5E9C8FA87A20B8A9D0DC07FA20353E72062F295A636AC44EC3"
01380     "64DD718B618AD64FAC6F37ABDBE9BEF09DF9C8E8F0FB40EA866F3566F978F907EF2FF1D30E"
01381     "F313E0B317E873CF6B18A0CB7303538167C155C2C0473B85194AACAF5F6FB6F7E4691CA69B"
01382     "AF6BC81C6A0DFBE8CEC7480DB342D452FD7284431ECDC1B206FEA1D831062F3842EF08DB1D"
01383     "EEC2F51064194C7C0D764F2CA7A6968E5065341C4E6F11759E96ACCD5AD736FF6F0F471E93"
01384     "3A288B5AFAE36C1EF1F117D91BD72B1A33B8F68CAE179EDA81526BA4187AF6C10776308D17"
01385     "9089DDEC292DF871DA31E08E738041A3B2FBF6FE5B52774A7D0B6713B8ABF0A4A5D9A3E2B8"
01386     "B8A219276D28A5EA967AF6C95E1C56D7B7E3E877AAEEDF70B6985E35E4E0811C02EE4B95E6"
01387     "4DA809A0A378A41399D504C4270173682FD353C588",  // PRN 38
01388     "1327D0DF4F98FC13CFD8FC87EAC26CD9BF52F8844EF532BF7194E4F4CD8DF9158FBC6A11D9"
01389     "A6AF40901A6FB4FB82B205364A878A4330 40F9F5687769C7F1CD3CB8F5BF9D05526FC5F7A9"
01390     "52D702FFA9CFC6D30508F28E0C802020944BEB591297786344DF8B057092A4082DAC163E95"
01391     "AEB74ECC623913CC3226C095B9D5FA48185BF6CB515502E9083E1FD67BBC749777CC98BF20"
01392     "5F35621523D8DA72A674BBE1AEA89E0327D8EF6A26DFEE142C5CE70716D2DB232535ABAB9C"
01393     "619EDC560504647620C1DB5D91ACAC289AC2EB8805EDE5496F51BD13D90DE4366842109A0B"
01394     "4E05F1D07D03C719DB321806A9EA7ECB5FD70C9E80C17827552D443DE9A92772239FE52854"
01395     "43096B7C7276B767E48F7A6FC57460EAFCF6E12155D6427807E152D435E00E305EC7997AD1"
01396     "6217700C8B859F89F3391007DFE38AD6B34A8769644976F2C2BCEF4B1272540BB0ECB22C93"
01397     "CC5FAD1EB5A1B1178504F7A35E0BB42153CDF5E4184E63D5C9AA140A66863B26E6773FB38F"
01398     "BC3EE829344E55F8C7FBDCF6636656171898862 16DA9FC6E64781A195EAF8E2705DE5A9095"
01399     "B93E90D1A31EEDF0EBED20B2906E6ADA6B42522093AE922FA4A530BAD07E8EF0A5ADB4D49F"
01400     "AC1DC4A5A6DC083CB38DEC75EB4A9257BC3A9613FC6993D499F5BE62AA03A1395E8D8F566E"
01401     "F09D95A58A4C85CF6F406558BC3453B2C351DC290E55A356A3CE3D7A56EEA3CC586ECAB353"
01402     "20414036667FF6BAD06E471A1363DF7BD65B360B68D6CF7FCC47875488A386C9A77839008B"
01403     "596B3ACE279C941A48A6E751EA3AE29DDD423C454D0032EBF6E17ECE497AF832B9ED80D4B4"
01404     "FD4CA48CF2612EE39EE8804D5E19102244609063D9EF9B01106109F837A857C4879B01CEE4"
01405     "D2CE94AEE2C827BDB6732DEDB9A1DFD4E691EBE956979368B0E8AB626887FD05BC95AA671D"
01406     "1711A6B8E3AB37827A7E4CD9D0B9B4D596E67099966DAB4E25040D9059720F651C063DE8A8"
01407     "90224052EACE6A7570D36DABFA0B6BAC8A38BA99AFF1B6E1D8689FE22D8194F6D96B4223EB"
01408     "41EC4FC071C47B4CE1845C40DF584A18668BDF1A00A36CA409E47357E910FEBC3FBA4DBB7D"
01409     "52A7976E90D80B9B0F0BF530456BBB0B328FB05FD223C65D24BF41BC70519BF6C599E059E3"
01410     "51E33E12739E198803FAAA48AB0401826BDC7168236DB0301587924BD2AD5A65E148858DCA"
01411     "1914A7A6E5C055AE6CEC8F2B7C114EE2668D990309BE76573AD8FE078639497C7D21CEEEB3"
01412     "585AEC412B80B6395EC69E2242FC348E2A6FB222D3A9ED5CA97D71076C190DDD0D8078E030"
01413     "0A202B342030D015AFEEF2B43D65C29CBE8706CBC341FA08D69D2DCF32AC9F5E6387060B10"
01414     "CC37F8F30C5552DFCEB1BD3C79E7B47B44A80EC57A9859BC7DB605DDB86886DA64A2C8BF5D"
01415     "AE05B4A01DBE4879053C8E1A38FFAB041F0B5405FD04AC8AABD57FA3A46413C21C320F346C"
01416     "7CFDC19B631B29812C4E215288D6F90708761 67A7AA68FC4862577DC02E9FD50FB3A3FB61A"
01417     "5530E1697BE4A3ED6C4E322242EDCEF457923B0023F6712C72975AF49A1E73565A4F17552F"
01418     "3C9B4FE16072F65568D78C51E4BACD6A849C20425F32185F79E1E301FFAD704E2D885F2BBD"
01419     "CF152A4B067B4B382AB97A307D92EE1FBFC1C10AEF5090CB8B5BCB5227FC3207584D0F7223"
01420     "916E1EF2CACD7C028827B2CA5146D33965F9C4EB431139611B6FC2427E5FBA87D968BC853B"
01421     "DEED3C33A3788B4DDB59576983A3951BD5D85779C76419C9AC0A54A3957CF3C0974F71E93B"
01422     "622AB66CFF037C91020904546672DD0FEB05BCD490",  // PRN 39
01423     "EA921FC3D33DD155F48B3356D78FBC9990FEB705BB3E02295DE80D7B39FCEA061341D7C410"
01424     "803C2844243A1355F0B6F15D09AF71D12A66A24C5C152EF830B95AD3C0F7B8CC8DFD33104E"
01425     "5C6809E22BA86B45078456A848192784A85F4939D90C9F53BBB31A7141F91B5140ACC1CF46"
01426     "34DEFDC070F67C4041F15796D7C2CA4E7554B12361F2FEEB8E7A28D0CBA2F1A707C149A30D"
01427     "DB3FC0C4C964D9A8E06BF872940C22E6DC33058BC59EFB6ABFE6B2E278084ACDB53CF8BC2C"
01428     "96AA58309730B2603045510E19CD315144F51ED99419257B5B29AA177F1625AECF95925947"
01429     "DD67C5FF6A8D704AD3A2DCF456A78FEB2431CE62D037BEA00FD8C25DCEF37CF5579A2EE1C3"
01430     "F1465BA89EA2CBF2EBB412907B64CFE0EA7624D8A839E733912E7F98EEA283C87AAB455E51"
01431     "7F8F479A7AE8B4DE47E22E0AE8DE0AA71B982D321A3CED0AD27045A14D202B191953CA21C7"
01432     "51230321E1E4AC73C3049403A2D423A7BC8BEFD783D5D3E5BCD7CDB75F8EA10205CA5B5B8E"
01433     "CBBC8564B30EF045C0B3215E768DF4D6420EA0BD956421065667E24B52ADF162ACEB33D508"
01434     "2C211B9379E6438D7F00A8CC2D247296B570D779A34EDE8D16EF70669DBE7EF52CA48BF3CA"
01435     "6783539DF8A415D804D404F46D91743B417E4244E923A01802CF9F5F1ED27F252B991606CA"
01436     "46987F0C49B447821E9147355FB20735EFC48412ECBB558F04A70E871DAAED3A2FC8B7A23F"
01437     "F915C88AEAC4F0CD8F12C845136C4F623566A3BAE2803716F02AD5F3FBC040E58C913CE5B2"
01438     "8370C516E7C7E08FC21CDE8461DCD960138723990F50DB0D427DF0634A490F99ECE15FBA33"
01439     "8C3967E4715FF15578538CDA1027B664656364D58B05B8D3A06A4CEAC3F7B49AB2D6540F9A"
01440     "54B4E3CFDE20F44BEF251D5565909FEBFAF44F5A23759BBFC4BA6021EE7D7B3D0B5C039535"
01441     "AB331CAAB665D91DE334B854B23CBAE5FCBB9AE2D44E527E6B49BB0082A0C3EE960D251AC7"
01442     "9CA0CE4EC8DA79349CAC602276B51DD37450B95629CC7120D08E3BCE8909D17BFC5EEC278F"
01443     "6BDCC95C2708B2064209B4D499A12826A98D13844A8DF00454AAE910AA5145C3B82289CC03"
01444     "4E7C82E3E2ED2F8E23E8CDD5FC40F97FCB35EF5AADBB7EAA95BA760471711911995669F43D"
01445     "73D64D07D56C69A4B2B2CD41DC03ED9EC6A92241D3FD084225D38FC625192F5AF85D299FAB"
```

```
01446     "778CF6E0E0D3FDF3A7F79942085CA592260C3458AADA2F776ABE79F372AB0BCAB56A8FD04D"
01447     "8EE81DC0C16BB569099114B91B7DF5E56C374111DD032663AF48EED3765CF698E675B930B7"
01448     "6475DB9AD8256CB7E4A52C5BA9FE2A3C5808AA2A43A013A9D2A1976C4645952B43BE206179"
01449     "863A740FBA8239AD363440ED9CCBAE423F22FCDC16C48C65EB0F484F38EFBF8F1E1D7863E9"
01450     "6F6B82F04F99CA434831DFF74CF080DCCA900214258CD747E96C27F1066778A5249576B7BB"
01451     "12FF31BF8A465C2B04208233A1FE0875031264DA6FC8AC54F686A7D40DFC65E2EC9A78A8BB"
01452     "751C072196FC9C238831BA2A9F798D7D3D0DE08A7FEE095236DCDC8B9D77EEAA0529C83148"
01453     "5CACA590C2404D4C3FBB36A590D9D6576DAD650D4BF6A8609C85D9815314603C365C07AEEE"
01454     "1FE424C0AD100780C8C85D8397A18C8591C26D2C2E48C64582B9FDC534220F12189ED2AB30"
01455     "A972A647BB60DEA0724D4BC12F64484B647DA7097F663BFD0F63A39DC729186873B333A92B"
01456     "0787F4E8A335DBFCAD7E399272194F9FFB6ADA78E296F634770F69FB851E583EEA8C9CEB54"
01457     "97A9A643044C777581F09A1B3FE6DC46290F7DA584",  // PRN 40
01458     "D4586907EC637459D2E42A147B17CCBFC1A79225D518E2515F2C89E4AA560FC958DDF8F81E"
01459     "726784B064F97EA6C6C853372225D4C8CC2B3F5CC2CF46173E83D09F68A5E3CE78A9D2EC2F"
01460     "7AD0E1731D59A86F44121C0455E850FFC48CFDA027A93D74C088C94FC94F77FB04F28792C5"
01461     "0F714387DF767B884D00C95C0F07E13241CA878A324B4F08986FDC52393E1CAAF82D298EB7"
01462     "91FA9DD823A12AB51E8C9AFD7A2D0A922F694BCC931F62916A4C348DCF4BF2775488B92343"
01463     "4DD905B6DC97AD331355D0F442E44440ADAD1B7B1D630A8AF4330CBD9FA96693F4250C2D2E"
01464     "264B6EE736E368EACC2CEE260F1D7EEF4F8BE70771F4C3E31383268937E722E40A930406B4"
01465     "BE8AFED2D8138975F3813C6546C5DE1C72510203E0540E55D4EB7A14979505893DA5545458"
01466     "DF79781CBE2EEBE5781D1A0B0F1D1444FB1C5E600C6EA92814E2DF764F123E015A1DE147A3"
01467     "F472C5D5ABC7D0C2DC631765EDF13769ECA5489AA0747338031401D56431AF6F8F0CE2411C"
01468     "C12AF4F8C1D265E4FACFFC71F432388A39D622264964B14725C9858F8442E0781561E0AC30"
01469     "3A79BFA0FD5DC07B71561265B9DC3ACA8C68FD7765B504B78963FAB0948EC9B90407897A1E"
01470     "4966DE16CC1E332DBCF44422C040C617D15671071359E93959ABBAFF81AD147C5359E3734B"
01471     "EEB3D4C904188762AB326B4CC635525070479EDD06B67772A652F467F5E95A10D79FCBA8E1"
01472     "A8306D659D0EBD39DF618F9363AC3F96E5E4BDE7B33A4393CABAC960CEF4E5E34AEF678E07"
01473     "3A0C87E80F357682DEA11BFD6D6AF70AFF4C7F3FC6109891E8C0AFBB24136F2EF7C976A9F2"
01474     "911FE14C87E5F6C09439417118CBF25983C23879A254B96613F61F56021C039BCAC420A120"
01475     "81A0D49E6EB460335620AF6C359A28FA32CC1EB2277211739DB1B002612DEB2B787875192A"
01476     "4C41236EB20B133C2781F7B97712B6D9C399BF21EBCDE6B09566086F309373F56335F4039A"
01477     "100B7813B2E048B90258F880D13F7F8B6BA20B4B699A2DA8E983D1E2CBA039037B1B399229"
01478     "37ADF1CFFEFDABB70F1EB5028AC1A40DEE89E29F01031F1ADE0A80FE46D8D9CDE478311AD2"
01479     "1F01B79F67523EC39AE80FE40D5EC2FE853FE96A4267C5FBE26F1BC3E77975A4FD3BE9D295"
01480     "2B6D5EB0A894B4C9F0A594075C9114E93844A0DF7FF53917DAED6E30A42ED59B704AD4382E"
01481     "85CD0A2FC77F37F303169C22B0D79458085898F47A8AB0F1007932EADCC62F0D18528C48CB"
01482     "393E94012FACBCB00F692A85FC3E1A5F07924847BABB7043A7BEFAD5F7920DC75DBB1F8F97"
01483     "F9B2B4932AA3D1817C26EB85812B486BD7E3FB0A9647C57713C5412D564B0E3002D7DE3DDA"
01484     "3F7296879CAFE0470FB6CB81B67EA98F43DBC3024291212B8849953D5ACB20B2B5EC95B9E7"
01485     "5091C6F9094E58302E87FB7DB179F56322DC25273EE63EFCBD6BFA3DD57DE04FA3A6C3DC46"
01486     "8B3AA47280DFBDAE95C1822311D42D33BDC0FA216773E8E3016F3270478819D9E2B532C697"
01487     "757547CC0B4203EDC24F2600D37FDE321D69267B45C5D4F7A52BD9DF858A11BF649A41A2D6"
01488     "191520D3A29B66812A02E4E67A13B553707792332F81787D53B2C56176F63FEBC81F842460"
01489     "864761F9D90AEC7A990139D057E0A82B7C17F30A37315AB2DF7D76CABE6BD748E288AB6F57"
01490     "6744581488C0757F6FDAEC4E3D34F6AD5AF0616D7436E92BD4D2CFEA92B586755AA582BCAC"
01491     "95A392D496F7EF7DBBE4AC50E8994D27FF3A503EAAD5EF1F9B7A84634B8937601880309505"
01492     "1C0D5D165F9393E3B84C511099A58D4D7CADA3DF20",  // PRN 41
01493     "3EB98A46F91DE7E3870144CEEAF95CB692B022D2E19C84C36BFA9FF42EDC5C413940812629"
01494     "87361544A4CB4ADED1CCF52CE3BD731F4EE873CFA0FB6900B0348397F86111313558F297C9"
01495     "B82CB185DBDEF29AC2723B5E0E6F24FA0F456A126899E99814F45C40F2278845145D8D8445"
01496     "CA88BE687A3DED47604C121E82FFC55E9DEE0797AA7CD9F9F5448BE6E200119DC5AD582665"
01497     "3BA1E3ADB8593A79B466C5AF053FA32D57641F51EFB15DC52F5150CBCF28528B8B1F3FCC05"
01498     "57EA9E2D6CA7D84ECAE8E9FCD8C0A7FDC9451A0C9F0E470DB496B1BAD754CE861F87CD77B8"
01499     "00D7962AEB8B5AF5347EE2247058D7DA26CE7A926E7F8139EB2D5B1DDA6F5C59408B48D886"
01500     "3BD3CE3294B13B3FB35D71A5B6C057154D7657C87EC3360BACF69EA3CE7D840FE64338A91C"
01501     "0B2E6D568DE270DF5920AEDAD1016150624CFAD6098400EBEC2092079E3053E2DC49666B36"
01502     "AC49273B5D1E77ABEF9E5C21A30740CF91B42443F4FCCFB4B812768A18BEF06F809CFBA221"
01503     "549B7EAE39BB5989A8688F12321E13633C331BFC529B69877EBD10053E7B35F3F35A1D62F5"
01504     "25A5FDF014BB74E7FFFB11564E776FC653067A0610766C34B7BC4C562F825EF73AA1CF5B1D"
01505     "BDD08CFCEFCC8326F9E6B07D231EF35A84970F81692E8805B1784CA6D79DACA06219DC6F2E"
01506     "4A00A12B0CCFF3306071EDB6400DE599D9B5B661240C652768BC6BFEBF1DD7725291FBC2BC"
01507     "424B2F46643EF37469F15441AF27B59DFD16D2EB31B45FB1DEA9888CB1F96C6FF49918706F"
01508     "39C301D5C8F47E647A8DBF7528E7D23F51481B15F69604C72409E518A9FBFB680EAB4866F4"
01509     "5D81AB651B45AB8E0E7E0230A3B475C147B5466579413414 26D20A7AE124E622E18654589A"
01510     "DEA5AB06D643D1366198AE4CB77D7435BB61C090120104D56FC19CB3294AE25C7F9469B45F"
01511     "1F68D3FE0A7923683A0C000D022DAFCC6F8BDE4F739CA0B586CE55A42B851A812EDE84431B"
01512     "BD50DB77EBC018DF85AC0C129389B7C02A9CF59DDB6BAC42028081B59C2AA25AFADA52D471"
01513     "C49EFF6535C2AD8290CAAF4771844FF82F5965F154F3B1E2CBB622B50D0636597F9091FC5D"
01514     "B3E0377E8EF33AFF204B7B0D072BE2C7A35580204FEB394D3504693B6284A441D121D0A91F"
01515     "D97432EC7D4D2270BD37148065C46E568898141E11C36945FD6F547DA97F38EF7F610CEDB5"
01516     "D3DCD2A7BCB2754A5B443E21DBB0ACFF71DB4F9A787A94185D0C98C2B2654E2430A3FE43AF"
01517     "4B9E816F0C3CCC13DF612590648C0DFF408CE967965D38AEBF813431881A051AC9F896E508"
01518     "2238E88B52DD95CAFA59ECEC1CD7170D08879C6AA1ADB4D5217E587A3AF1FA4461CCE3D4EB"
01519     "7A3D82432C90052335366738A60CB25B1B5E90D2B39FAA80E3052DDF2BEC9358868E0F97E4"
01520     "FCB707E15FC051D21810E8A38C6CB5B0FF05361F225E8B8D10C958C7BF84EB31813DB1DDC2"
01521     "E8D898058243A9A2F41E3B55B33B87E832ECEAF80DBB33E34C3F830660E6E228586FB6E825"
01522     "F874623EE4BFD8E8FF1736627881A3F1F06B498EC6D5FB0CEA203D5C7C145EE6F09B0860DC"
01523     "D86436B036C7CFD873A245D976C63BC5432B487AA0DB087A62BFAACC6575E265DBAD89D41C"
01524     "FF89E626ADDDA66835CE497C17F1440371146EFC8846C7FFA13C4D446078324598E59A68D8"
01525     "DB15F2D027DFA3DD2E7123E380B63F01D8D07C5B42B43AD76D95BD7B57A5CA2EE971F1B9BA"
01526     "4DF6F8DB0C0E1D02074C070A745FC09D1BFE1900D1820CC42F66FB97C39BBACF3C66FDF94C"
01527     "09F64FFD20430BADDE7A84C3EF0D82F6E50B96EEF4",  // PRN 42
01528     "9FDE160085FE8806B43B4E731B2026CE6395293C827458A52B4F92D943674A864C1948C7FB"
01529     "165D041CDDE90CBD1290D664F2D4DA43674073E420BE61A4D4B0B20546A64FE8B9B1846226"
01530     "98E67660C364BD2F97CFC5176DF4E84AB5215F02DDC3C47FBEAA523DC8CB14193885850ED1"
01531     "03109C3EA9D49422968FE54CD66D29E94187D73B446C21F369D506D6178C91ABDDDFC6736C"
01532     "94C7C5C8410AD4C8DD6198928FAE1ABDD426402810E6FCDE33D835C40D41330642EB2D7FEA"
```

```
01533     "1480A517E39B78BD85668C2E0B6B1E30AA9C1614B07089577A20BEB06FD51F5D73FA78FA64"
01534     "34E7A1979AA37906E421EDBBF9A2097202F5A63CFF2AD0061EF94BB830689EB5A057AB1962"
01535     "1CCB551D75AE0BBC5F031C7870F35CCF8EE8457171911E198F58AE679283DDDC57AACEADAA"
01536     "A0C5D0B7E4B22D68C2F9DE42316643A720A8421FF8E6BBA5BBC639BB235B2BFA48F69E973B"
01537     "BA14F30078B178CD67FAB150DC24390E8672B3A92B5FE92D30D70AF6A6318E0FDB466F3CB9"
01538     "FCDCE58E48BD8A37D18768990C4424D4AD0A7A269BE56A7C58F6191B797AB4E019C77EECF9"
01539     "B79BCBF74616E7957BFE239E47F6F9AD9F4FC1FC16BF82763F34609BE72A4E6C90C18141AB"
01540     "A71A16F96C25AD4385340FE67FEF3A90D6436C3455D39BD16C7C4E5C1365B3FCD110BEA8E0"
01541     "D2FFE63A3DF63CEBBD2D7FDF7680A67C83CF51416B4AD115E7054A5CDAA6443476E207D5FA"
01542     "24F87BC438A30A5B078F24AA5F966DBB87CFB2C3FBF63BF84ED3CB4DFBDF60D64AEA1786A6"
01543     "2E80176E3D62AE69059086D3B3D71313A7B0DC288FE63226570983C85D857862716B6002B7"
01544     "2DEAAA8D4842AB531FBCDA7009996B6A8788E2048D7807BE6D1C55C45F8AF816F0D1BD633F"
01545     "80F346AFC8AAD1A350E3A13FFF3B1D16F600A1985AA43C5376160150561FFB4333974AF3DE"
01546     "6F22B24213F35E101638C93B216F5EB484FEEEC67FD38F4159DC521A47AADF8E42B6F1EC43"
01547     "DB52AED253B75883393577366084F90D6D55A56E6DE457A3087020DD90753D84008EEBB883"
01548     "10534B4DEA40EB4F5282AE9EF2FD20EEFDEC1EBCE713629113B1383827A381BE262F10F097"
01549     "1D45C2597FD9A108A09E32DACDC81783F283C44049D13680CC7C07CF03279DF86A22D7FEC2"
01550     "42A7A265DF0319A6F3B703F84310B62282C3C46A9E19E81B474341D4CB16F763A2B78FCC4A"
01551     "56FC128048E8E61CCF1DE77023048150FB3C96D4A0FD7FCCFC8855219A30DD2A3469D5670B"
01552     "465A5D600C645C802D1479298DFF8818C152A21208D520C301E7FBE5C55F744E6AC45F56AA"
01553     "31E3A596B31AC4836F53A78C427D8B3284A34B77B38B6AED44362AB36F72446A8EA179A278"
01554     "45E93607C3F77F44E420AAADEE877A90AB39E5257B6A5A801A04B3ECF2B78BFDE9089C5131"
01555     "D6C031DD3D02EF4246437502339139B5484C6EB481C3D3337376842BEE792F148FA3DA940A"
01556     "0208AB3AF8F6F96F02F64C42EF0D0FDBDF178493BD724700A65BE1165FAD8706FA529DE891"
01557     "A95B27C0D15C8C3DA300DE05DFF7B03F7D542C1BA5DE76CFFE23CC9A265F69B8D0167F25F0"
01558     "6C5C5918554CB342EC80CE1B5D671E99ADC2B1EE0F93DE52325CAC69BBD25647E43E8D30C7"
01559     "96845B0A656506D9BAE8B9B43F1086223020703B789E5F34465BB2A2E044CA2644998AE6B2"
01560     "99DFBA547F0DF087CE37614088DFFA6D1444D6639C3D7EB8FBB6518B3B6C2E8C70BBF46D53"
01561     "E982CCC0BBF6648E403BE6C7C50717BE49BF4BCEE20C57569B41B40C666BED36B7E986E91F"
01562     "72765DF50E36C75544A766E0B83CD899C2095DFA70",  // PRN 43
01563     "2A04CABEA481FC7FFF3DC02DF99CA09574A05F592C273C050A81C787EC8B7DAB7C42C553EF"
01564     "9FBD8A8F9EBD77E0115A205194015E35455E80B48EE8750C546A6BCB6F782F1D6231386B1C"
01565     "CFBB1CEAD8CC6565DDF39995B4E8404FEEA9BB1EF537C67C7BCC7EB820DAD3B1708EBC9CEA"
01566     "451F6F87B6369615F39F0716719BCB7E4799C8AA568DBE32B3B279C1CE3B48B205D23CEF04"
01567     "1869039D9A2469D5C5A757E8C3FF4978E50A4FD149E85CF51D9C2942F8571252B6D76EDD00"
01568     "CE538B2F7FD2944709BE2AA58B801780EC89B596C8FD2C938DAFFBE05B226919BDFD56D738"
01569     "5F8F0929001D3BD96C87313B98BAF1FD1F350139FC32A7E84351F052868B55B1960D192FA2"
01570     "0F9F7940FC75B54031B5BD73DC501C86F253BA07B0DE481D97F251F8E2789D47AACA0AA654"
01571     "730729971AD5BEBEA7F98C32B3D61FE2B84D5535AEB0D38DF4050CD886D7FCCCDA95480B3A"
01572     "7657A8EB3C518FAEE316322CC33A4AAF71AF867BAFF4DAF11BC70E8DAC32CEE8A6E9B2DBEA"
01573     "A34CAB6DFF3973748F1C24D8EFAB30BE1C21BDD33375008FB39979A9125A3CE7544ADE5806"
01574     "99CA17533752F483970B3F5F0240ECF71CA97F808950A85CE993513889937F7EDF5F54B95D"
01575     "4A1D09044E210E284C0DBCBD6DAA44546C9C9812464A166418A1E45FFD83DA719D65FD313A"
01576     "67012AD8F710B9A74DEFB72FF688FE69916C0E28243755484555EB1A81074610E48FF80C25F"
01577     "13EE7BF098D5C4844868019511834BB8AE6822F1988EA6EDA00102B679756CA8B560A75F62"
01578     "E74F35B2FFC31E0490760C0F5AF6E6769632D2C5F479001DBF076453D3F56A935E809F2376"
01579     "E8ADE65CEADE115E6A41F294BBB08B4A15DF94EE426B0B02A367FB261A0D3D321CFCC06803"
01580     "62C0A1B4B7834873050C2FF42CFC943EA973BEDEB99639FB1F7D6C88E024518D7C99DE7886"
01581     "E5E17B5020ED44F10C7842766B8E13F485A2ECEE3AF0AB60D549A420B166B8C5F51988EA15"
01582     "E47A6B002E1DF56CCB7E6BE805012A2CC5D00FBF02D905537B99F7B972945B4444DA4DF442"
01583     "02E5D406216F5D50E8100A099743C3A0788B6E3B2747861D7A36945C2613422AD4875D8BA6"
01584     "2E6A08C60A52E93249726657014C0D91094C3EFD4109D2CBE40214840F49D258FFBDADE593"
01585     "9B73DB80D89965D5DD219EA58D7BA8EAE202BF9CECCF9DAB3F1685DA45CD948E0ED96E6DCB"
01586     "EB179BC1577664E4B3E1A8A6D645A08E1E481C7E2902D16C79A7051F8D57DBCD2A66F45FCF"
01587     "9F255599887B3E7343FA48B1D32994E8549AD88348443DB287C51C170BC2FB01DAE1EA09F7"
01588     "DEE8EF145EA61A48EB94A4B555C79C1742ECC06269F9D5D819D3C1E8AFB6E1954941111096"
01589     "E4301B7FB8C33C638FAB53B6208FC4A4146F183C4FA61CABF9E1479701B555294DFC59130D"
01590     "0FC3B9EBA0F1203D7A1A87D38E2DB07E469A25449E1AE16B3B84BC27E6E67BFF93D3BE05FB"
01591     "B2323E566362D94694B6004E5C536D9600C7B2DCCF660341C035A456DB70D4048A872D3EC5"
01592     "79773889A2776AA85FDA398B91A5269C60FDCBF33B789EEFE9F5EB5ADCBAF1F3C6280BE20A"
01593     "54D5D0D62B9259A67FC97A71424DCD4E94C0CBDBAE07A39B3F4F81B521E375D80484BE055D"
01594     "3296BFE5AC1E752A3DB73C3AFE2350E9316622285AD2DCE65851E392858939BE3B2FC25CAA"
01595     "2C0016E2FF66086E0303B8C681B84DAB2DAF1DE22B36E6BBCD31F27F233FD08D3399EFEE0B"
01596     "E6C054F8BD1EEA98823EB7AF305916536AB0EA077F5E092466B4417C97CB6789FE34AC1BA8"
01597     "1F3EAABD0A9F4C8D3F3D24D3F126B392CA4B18D3B4",  // PRN 44
01598     "3CA56F1313E324915404FC17ABB1C2BE7B841795544CB86C08ACF2AC69BE7DAE0C4A644763"
01599     "0F3BB8D84656ACD18C74FEEC701806C9E649C942245FCAC57819125D7834A9BF749A7DD7E9"
01600     "847AD34D855FEEE046F288C72DDB440534D494324BEFB896315E232EDB57E9A4ED4CA39DB9"
01601     "D72EC99907A690D6FDCCCDB744770FB2F8B3B57C7BAD559502E04EF3DE53035D3A38C8CC67"
01602     "8945A265D708586CFB2501952B920E1FB4EDDBCF2019904F47E95E7B542C33D1B338B71659"
01603     "18926A786FE8AF2A86089B1E0D06117742ACB1E8B2F6CD607C90A0355FA5128C6FD7FCF0C6"
01604     "E50D1D9440E56645771B49071C59E298E316E72981B96A53FB979129DEF897E6B7E75C2DE6"
01605     "7A5961CB63BE8F360C8B7DE5092D6B3ADF79B1B143A5660E8BFEB5A60A240A5E94E65DCF9E"
01606     "8FD25E00B7BABDAC300693B3111DA565DCF8B4D97FBC4E23DBE163E0A42F2E3181612CD80B"
01607     "6EF8277CEDBE43A59844A1FFD632B9CB7F8F5C0C60C926007C0C487B5E15E2D3FFC672A48E"
01608     "6A2D88A51F095109DCB055197DECF67EF6740FBD63F5A6D6B2A94B646BD30B981493E39057"
01609     "88CF8EEA2250E7F6DB02FC09C119BFB7ACD45A4C26CAFFAF820D6801C3C9F30878C0F6C3C6"
01610     "EA9D565CD122E5BE2D21E444BEB5ACB64ABBA1E93DE184293E704C2356CC8695948D2BB76B"
01611     "90251E944062D163C39035C73A5E94F9DAFFB10DC9F89B553B488AACDF2F4D0B51B7CC4B6D"
01612     "1CA603D9DC1453B38609FF4C336B296FD0DAF8F0376D01D973671A93E7AA744F1A4EA1834C"
01613     "8743EFAC70137F4B6873008734FB864B9C5232DE56415B15AA42C786ABF2DE1F6007B9563F"
01614     "9166118B3F1DD7DC40BCD5E0107A56FF3602B4FCE4446AECCA6F360273A1111315E86778C0"
01615     "58331DDB5D206B9A8568470402F5D8092A42F171BC10A0DDDB1532A7FD4AF79C5D103B861D"
01616     "07A18C9DA515E51C8176F79D674E5D4032AC04E5AD6384044B280DF9199F92DB83C8779634"
01617     "BA6E4601A5EE0EB381E946765F989CA9DE204766BA9005684BF6DA5BA8C2F42DF2A6579A39"
01618     "020685A375D9A1BBDFE02B90C2977D0486AE3D45D75D2F27836AEAF4AC298AA6D4401C7219"
01619     "0167D64A3E35DE0529128CACB2367F803D05F98788D9D024E0D88F8A8E02A8566E46FDCE44"
```

```
01620     "DEF16EAC95A0944E67F1FA9A7DDFB0EE1C89EDF63E44A6810000006FB4916FA7D5C993175F"
01621     "DB8E27A45DC405242AA4CC3257FD3B0FFE430E6C574C7AFB448975A37D6C12A399B7F75278"
01622     "53FCAF1020D514E724637EC59152FD4BE8A232AA7AE574DE8C104F90FA4661E278A531CEFB"
01623     "77E6B63EEED1DF950627EA67323D0A9FDF002F2E93DB0C6AB0C2B8F1D9BA0F2BB05FAF1089"
01624     "C403DD0493A028CE7F792B6C85C1226398374FEE446C6F1FCE1EF617B75C7E3C398FF204C1"
01625     "783824619F49A49CA9D7EEEA9377B619A9CC8F5D20C7EF89D0026D219B591632E5CA909986"
01626     "FEBE7C518A0E7C92D3136058D6768529C67BD84664E4FE4D34DB662F0AD738B626E65C86A7"
01627     "F9090042EB11816C4E74D12E07222B696C8CA460689CCF97AC81CFF14167CEC153C1112700"
01628     "48B5046DFF64DDE18DA9E6EA2F6C7AEB0DD7DFCD9FE37D823A53B9F109C5F600FA9FAD1C21"
01629     "DE694335AB817951B2352DF7B67A19F9BA7C684306DA14B2395CDB3F1C1BACF920FEEAD158"
01630     "E87D88EF665A9C3D3A43025BB536CB285B5012D3A3D81A827B1AE89B4977FAD4F5E8249C5D"
01631     "6229A78C031415A9F3D1D5A55F2914126F7E53986C05B5600C526A31732B3EEB6EFABEF5A7"
01632     "C707F057ACCCADAC4FD9447BCA3D5CFA20352AF8EC",  // PRN 45
01633     "03928AF2A832A21DB2E2101BB6EEB4D135C6311C6CA3E15608F112373ADFAD87630BB67809"
01634     "0A41506810F32576B24789B67C9FF8DD654956934E1290941F7C55BFA8821794EBFF46B5F6"
01635     "2B28555EB2D34FB60ACA9BEF8149D43227A4A83329ADF75FE42CC48B154077A3946175E799"
01636     "B698300238F7A5C3389E414462B987FD9587C2FB66E873C12BCDCF897423FD6DA964BC84FC"
01637     "F1A437441C2D0E62039EBC23424842E0E63CC872BDF00F0B1CDF0B597B0F1D102EEF68358E"
01638     "D8918E7E1727CA46E49D2504421D45DE9E744063B80C8C1932E626CAA99504514AB5527EB3"
01639     "87F4936B335421B2678F737EB17EC133A3DA1AC354FA5D885A4BCC5904D858515A5EF2EEC3"
01640     "A9D6C85AEBFE721B9831F511A52FBA7E626E4966CEF6337ED45F2B3BD26B306A69700843B0"
01641     "B8DE0E0F70A3D480ED7E1186CC0917B53D461557524E245F51CE6B4E20BA5D8A26877E6528"
01642     "365964F0F729393B48CC5FB3C5A1D46F497BEF2E824BE03CE7CF68D8D91D67D72FBD6D67EC"
01643     "0902153AE09351FA3D3F24E93FD55F75DF561EDE2AB8CD57ABA3C745CCC6B8AA463B00EA72"
01644     "9971D665919154D149889B21D64A19267C043493C72CCEB25D8A5EC4A0957E417D8BCE0DFF"
01645     "84D786C1EF4C727F72C7CA7952F4E1584AF38150EAD2664249F8C17F93A53A099FA179AEA9"
01646     "CDE4E5936B3F85A9BFB41BDBB64B939AA712AC420FBDF051A3132CB88EAE09E146C50A8EF7"
01647     "41B57905CC3438E56ED0E90427728892EDFE685D857FD196CCD2B62F87D0CE8FCE5B7461EA"
01648     "51239FF6A1854387ABD9B248F576E1BE9AC34C0F2CBC2B618B9952FA804CDB7048956BA184"
01649     "7440EBA306BE3CE42084AF8851757857EB38F661AC4A2376503DB5614C8CA4C9C9564607B4"
01650     "21781B5503C2D71CDE7F7272675B8B6D8A633F24E2BF4CEAF625692A695E7E7CD38378D266"
01651     "5AA2466B90733098F754E96E37E00B549A2CDE3C634A5A64BEF38D4510A5C586E9426C3EEC"
01652     "F0247A5D2737121B84B2CE9243B068ABDDE5214A3F7814AFE3BED1623AF9B222011F48783B"
01653     "356395412F7E575948E6358F53859C9B47745A3AC08498FE9AB2E70E304106081215C75CA4"
01654     "1C7F108544798F56EC23A0A34B6F7120162BBD9B0E2CC605B42CB3C92CE87285407E6187A8"
01655     "247DB60A1E207F47D7B7EF39FFA73D4BD87E62B36E7051F744CE074199CD04CF7D7709A91D"
01656     "04068178DDA51DDCA5B90C9552FE2378101EA6D7BDFA993040BF2F114BAC01F7B5EA7E89C6"
01657     "AF9219B12B79B7A76D1FC53AE793B21DECBA9855E5958F9E17ED2F54F1B8B2208185CFA968"
01658     "A183B66C9E536FC943BF0F70186B5CF7F74FEC1BF07E80BD8650DB2745BBC368EBB4EBF07E"
01659     "A7873187EFBF36BDDE8EF65957458950441C5FEA3F666C5F296359CA500F21AC776C458038"
01660     "3B66BF2DDEAA4B88C366C52D66DF26735A27E03D92ACCF01D74FE0812FA63863396EE1DD95"
01661     "E852894D4B4BCA87899086E6AAE96A03ECAD7F77C78A477C2694F8FAF221D5B1391B7E8204"
01662     "92C039F1B003C4128ACE0EDA2131B2E4D9F879CA6B7229FE21CDBAB2D8BD7B3F9F23DBC185"
01663     "0445C6A50FE4D5B998BB45B54623076F507584366B29AFA7006DA4761E60C116FAAB0D6AEF"
01664     "0126A41094CF6C77B11C8A6784196164AF92BA8AFD58BEBE47A3518FC5DF653602222870E8"
01665     "85E86E65BAB3BDC85F728055584D541A60B81C3438566A908A35036468B63F93B9398600E7"
01666     "6F5C5AA4E7A63C49D34382D3B5E6071ED3B028368E4032A528BE474C7817DCD889E1C1DC3E"
01667     "17521FD9671F6A25678637648CDE6EFF885C419534",  // PRN 46
01668     "4FB5B9E9C2DB1949B53FDCC8AC765E5606735AE1F566749B0268F3B2AB3EAD97D321933EB7"
01669     "D8DFAB60ABB792815BD22DFC08E3C3D34A0338BF4E7BD07982C2CA71E2FC8BBEA7F915FAD6"
01670     "4FEEA464990EEA25FDCFCFF77CB6C153942D3BAD6C14721C84F7EF4600F9A5E07EAB16E384"
01671     "6C6F0E664F27B80C0F82A16180E65003D6044875FF4B36F85FD7247325EA993E6925982A11"
01672     "243A115D65B1F93FC715B3A90B90181AF0450C1573C6F1582F7FA0866698BA9F3CBFD6D3B0"
01673     "5557EA7547F51C9D55F15251DE8358F788C555E42A28EA250724124BBF0F9193D03550C6B7"
01674     "D77ED523704D0A5E1262EBB026119FC1AF766492DF5EA6DD019428CF3D64935AFC1DA3E596"
01675     "0B08B2FF0B19A1B551F437DC25A71332FBEC6C0BF061A521B8605E1E9B45FB17AFED028D51"
01676     "B75EA7C66870DBD8257D7001E5EC472FCB247308C770CF49C1BBB89689A1C48212A289FADD"
01677     "4C48D5386179C51DDED4832C848AEA9B7FD85C867181EF081F2935F5E3A6FB01122DACE51B"
01678     "E5E7B9D18063FA722022922EE48D82B69C556509383AF26AAE533EBD536B52280586CA0367"
01679     "CC622938D09B0A7135A6552C187607640D8D866DD6DFC68C986C8318C3A5C3ED446AE59D0A"
01680     "A656A4FB0C4037B0975AF167CD94AA32F4485F467FD7BD3BF46FC8F195D7157DB1EAF531BF"
01681     "E65000ED01844E7F0835929049C4534BC2CFEFF0554D0639255DFBAF7C9995F03AAC74230D"
01682     "73DCE1899BF3F30D9936ECB98E3B61186987DA5A9512EA71522CCE9C6273B5F6DDCD69CD7D"
01683     "B11C2D921114A7F7B3C88CE1234E0177B9202C7A06651C49CBCA7E7A1855BDCC8C0ED401E8"
01684     "EAA4C32B8472E06EA18D152E5594D3364791563992D8C5219D164CD48390386CE4107D547B"
01685     "F045568543ED7B535F8AB942F177F78405974E37F123B15723EC4FB9018A4029742D0AD692"
01686     "31E16E90026916C34E608B2908217F32F1199619AA95C892591483FA1BB847E144D46EB248"
01687     "D660E35A830B06B9E4795575734F043AA9D44AF566151478126D4B0B89F5A0ED6E933BAFAC"
01688     "360D827129135581EA0D692735103D8AC41C444BFF0F4952F869F63049290ECB411CF8143E7"
01689     "8F45B439A18264BC0DC0E1B635FCCB75F345670AE0BCCF56A39C44FDA88EE0B19668710322"
01690     "77F5D79767FE48B286261BF9CD93455CDCD87972F6D7867587829423ADE21341CA240139D9"
01691     "F6F9CC89FDDEFB1E59E9FA53D5A4D6FE7039FC8C3A8E9CFB89289D5E7B78ED23C97F71B2AE"
01692     "502C091E216F376294E22A9FAC057D30A061DAD81DB1E342336D32CCC62E434DAAD27B7256"
01693     "ACB48BECEFFBBD6318C17AEBE07B905D1D10BE64E2D73FC58B04465BEB84972DF6D0ADF01D"
01694     "077BEE10695070A7ED156DE96730F489F92557701B85D13BC1622E483D42A6ECD2D142C8C5"
01695     "9373548F1FF0DDAE5F9E08F00EED398B082BE242D705FA6892D2759EA4E4E29B971375301D"
01696     "96EFC3550795F802D0A066AA085B204330E0BF87C2034E4301C1336465198BC924FFC9DBEC"
01697     "1346E217DCFE80C6DDDB44E9EDAA88C5E44CA81477073D6771690CB73214B8C75D6EAA10A7"
01698     "69A640F188C840E1A359AB439F846E5CAD3EC0789D5E09DA11017D2296DC4DADFCDC521660"
01699     "5E26AB808D2C51E20096DC9FED250C3608E1D94CD17357BBA29A8B943A07A56D76B7A2CAD1"
01700     "5063885E477D78D78330D0BCBEF9C894CDBB2FC39303664FE4B27211ABDEBCAD668F60BBEB"
01701     "F8126402A1873FBD651168F067560C5BC14477D5D0757FF0E9C0C1360F76E0305A17988A0E"
01702     "AF8E2E9958030F81D4F3D56C5BAA3C34CBDABB130C",  // PRN 47
01703     "101EC76558BD3A5C44CD26F555F7313C54093840EB0AF55F19260A5514A0816EBBFF4D7A52"
01704     "5FB48E98A45B0BAD7FAB31FA0CAC69A6C43B0AF7B6A3CD33348F0061F637AB98A1A0BACBF0"
01705     "A006BF432004C8054EF8ED573ED868D2AC47E3C6467C6413204E7AB54A07C893B98C3FA8D5"
01706     "E2BBFD920074CC1C0469B281FD949C265D1B72160257723CB8E84B0657A33DE45C2A0D43FA"
```

```
01707     "B8BB2C47B57BC7DB419C2547468889D50018158FBE94F82A589B1D3F9F5B155DFBDF4F38D5"
01708     "D5F4C37CD6D35D745AF47AC9EC65A13BB71E71B9267205F35C6427B7CBE90BFF1331E31D54"
01709     "CB8B30E3ABC8C65F89213A4A2A3860C10EE76953735907911D975C88360591F248670CE560"
01710     "9A8F621BC5B08E8C2F0ECD645D77566D5F02F972ABD7015A1C620ADB77783636B6498FB8D4"
01711     "337F85528A68BC8652F08D5581917723463A76C9EEF63F2FE84E58265A851C0F19676A5709"
01712     "8E6CE404DEAAB955C02D339390B75B3AA9653D1E785779E9C2284707864A68CD271699FA3E"
01713     "A62963EE6FE7D1B0A582A3509BF66BE479E6AE5A0EEA5114A74809B4736C3425E03A9149B2"
01714     "2154E6BB9D546A6D5F92FFBAC7EB619A09BEACD7410B594832035D1B96829CC6CBBFF4DA53"
01715     "B7D6BC133377D2512C8B77EB2B72463EE917292931959BB2CC8091CB958AF5E6E28B2518D0"
01716     "A7823F8704FEA6CF5E901B3777F72411CE4811E4D64D99E84F10A1FE1BFCD212EA1ACA0892"
01717     "16E7839634A1751EAEBA4DEFC2037230C5154A6D8662D670306E8155B701525313DAF7CA04"
01718     "9CCA57B270A25E6C8C99B9F4E61C8D849FEEA4356B516BCED68F582E31783CB6267E2A317B"
01719     "B6D08BB0FC93754F6AA84320AD2F6E802E4B1AC6C009D69C00FFD390D8A66D4ECB335395E6"
01720     "279A755283EE9F552745DC2E93E2499E4A8C1A9E2DFB40DC29160EC04FFCC6D6821D1C0BB7"
01721     "534FE0E56090983C3DD1CCDE18B5454B5C3EB5B2EA5935788C57B421BA641A919EF4A105A5"
01722     "6FCE207F5084AF286C39A858E7F58CCE05D69BB1DB02ECAF9F3E1E7BD6543E70B6DC019B51"
01723     "A0580E5F1E1F081167CFFB5F8C7B5A999CD0F66F114A34A1AB6D097818BFE41A955AF2F840"
01724     "6917091AF923273017579B28E5743539E0D60AAED6487C74411E82C1334DC0121160F407F2"
01725     "0CEA14F504D1D52D79BC1B0A2C1C34AC9EE1D079DFF59716CECFFFA2B4E89F606D31B20B8B"
01726     "5E1AB6ACEC60A391EA284974472EB1D6676A5A1DEA5EF22468CEAC43A4FFDBE6AA23FCFBAB"
01727     "1B14C7274F25E7FFAA8CBD5EF3E3530D4C2097361DF9C3FBE45ED0A7B1A6CE116FBBC7EE5D"
01728     "80E6922EB23A4664339C2E219C5FBE419105B9079F3A228A4D412D5F73DF3139A751E91A0A"
01729     "75BD2B8A34430A7BFD3506BF8FE9E2C1ED3CF68D26A141967370FE0782F70A50B2ABF80811"
01730     "86A6B629C990ADF48E08343CCD478B44C9A685B7D19484B9E82F05EA853F77E7BCAD705F35"
01731     "043D34EEE2B3022AE9038D9268635A2B71EF912F1F390CDE49CF7AFEFF4B08AA1BD27C0130"
01732     "AD679D63A5B586488339F86FB6BDBDB381E2FB70D726F3322FC6E0801F24E34AC97A23F029"
01733     "FAE5148AD5CB8A17963F7EC7E9A670AFFD90966C35DED303DF28A3426307CD11F5E6054B0A"
01734     "D43D0358C7666DBE824FC037E87E84D214C92D4ED2B6319727729A53924FAE54E48CE4C522"
01735     "E97C6A5C70186835D997265E29FFD68B5CA9CE5C52D90130F36BC5C8B206B157FAE7EC152D"
01736     "ADC94ABFBC9FF9721F5D39A3A2672054381D7975FB5B3BEC6F35D7175B82DC73F8FD425920"
01737     "97126775A28FD7275F733E65EBEC299749A236B6C8",  // PRN 48
01738     "C91D4F53C54EF5237894D424E84BB8F75ADF73B0F4D3C70751D432591C3FD585A78F29EBA3"
01739     "EF8E1BF43FCF2B84E9EB7DF6BFF20BEB1BEB3CB8C4B7897252E1452172EAF89AD1750A1DA4"
01740     "702AA00A532F2E90B2220AD9708AFE71B1EE042ED8DA8C3337985D75B0E250D8AFB24B6F2F"
01741     "DA32745C7EAA26F8972006EBE2BE4FF164190823A3C30FF5B5FB66DACD921E7BBC903E7660"
01742     "1CA9C8A87C00B428C6C3A0C987BB5F1E015E479C9CD87BF846914C0FFA96E6C584690782CD"
01743     "602B6FCA56B8A59D97C0A34FE50216FD8EF5BA3891799DF06AE38443D3BAF5FE5013F89E0C"
01744     "C4C9A8DBAD2048494B246B22DF66612C2DE8C89AEE0B59E3E062A9F8C7A51B4D01BA2617D7"
01745     "F7C79553C1190554549 48F57146D3D665740BC5B6FE7CFE6F379F073BF82562FAC75843A87"
01746     "3E9F490161A9DD201B4197420A562C8AC27214B09150C7BFD5BD395872FE7F4CB41DA9671A"
01747     "234127BEDC4F5D9E190AABF9E6DD65702EF88961A75F2E6CC96825FB78BDBA68EC710F5C53"
01748     "076F9FFC5989C3F0AB4EB89BE340FE4FA38B315C9574912C73215C99CC56F9D9E85CC43C9D"
01749     "886B6FEF42E86CC7112CA8A56DC9EF282ABE9D073F64928B45A050025B4F44DEF1011804CF"
01750     "339BAE62EDC5A651974A414DE8D214E06AB357FD3B604AD261CB8285E9EC1B52FB84D931D6"
01751     "A6BD9D539E4A0401941FDE961A8B732F043ED5DB88E895CE2009E352838CF443B33FEA66B4"
01752     "F69F8152B03EE6B44B750673ED3F7CC29EB9BE2D450F25464ECCEB7E3DA44BB55E47E754DE"
01753     "F4255988FAEEF63ED79B4CA007E0ECA2270635039A92418D40C5F433C00F47DAF5CAB3CFDE"
01754     "14F7F08B2FADFDF90602077EEB72EDBB81F55D69400E1E90F16119481D8D117A5505FAC65F"
01755     "967963477 76C6DC9CA4823C2F3CFF3D48FD10674E8085D22CE74812E6ED4B225D8535B5D36"
01756     "CAB4354FE6FF2F7F856B2D6A592A2A902F6333F94D46665B9CD64E7A93110B78C6F1038CDF"
01757     "403F8F65F62E4F972F0493E87981E535297D83EDB51A4CB0110CCC226CA53A1B0CF5643A92"
01758     "DD1B1204FACBF10539DEA656A69C2DD650758C29A008B3B99E43FC45B9AE1882B27C29B02D"
01759     "80714CDFFCAF5E079116DC26F47E3C9916E98C6A1017474BBE2A8326F05323F29513A611A0"
01760     "48C8D2679E06D71F6CA45844077A5009CF272433FC892609A94C99ED91FC51304396FF282A"
01761     "FEC7096F2002982EE445C3B2FE3423DCF11BC66F36CB531E141383C11B33879D72A16727AB"
01762     "8B55B48928E13110BB9604D4209A6B1B1484A4B2A67A35F5B2E4793349EE92287791C928A6"
01763     "5ADC0DA238EC4F5484D006829ED66A6F74883233251CD7326D01FBF2A26D1DA98F62AAE070"
01764     "3C9C70EE11E39504966C7E7F7A9C6369C69733C08A61A123D05C879E05A26FCF31B137E21C"
01765     "A45B14DC337045B73922CC6BB14D5515EC036AC01BD970E99B68D39A454F83373EB32D0E8F"
01766     "70B80E7EFE20371E4C054BEC3308EF920DAD6FC7EDAF01C2F40A03F249656246D08EDAC870"
01767     "EEFB37DEFA6DE812E38EE3479FAE7CAEA76453866D2166DE450B4B88F360DEAC68268FD978"
01768     "FB50A71686774BFC213E06D59D08CA7AE9A7C7DCA14AABBADA9A9A1269B0DCD9CDB7ED4F0D"
01769     "5EF15A531745E2EE1259B3139EF29B3566F6E94ABC7C1053923D7796BC43926BC6A927FAA1"
01770     "5DB29473C377DA06698CD366F8E68B07F71F932668E88CAE07F3676F3864E9480855F1557B"
01771     "F14173B256B3BBDBF725BEE677D286084925EAAC0E428AA1783401043B0EF734FC0C5912A9"
01772     "780CEDD5CB613BAEC37D623F533AF8C634974A3374",  // PRN 49
01773     "AFC22B34A5791F52DAA66F3DBF2702032EBC688D1DE2824845108266F95BEF050B363999B8"
01774     "3BD69EA7EF3E250F4280314E5D832C58F1D85E22EAB4255493F913AF7B08CE0DC2AB3708E4"
01775     "A3F779196B339625C38FA87B8AE0077ACFCC9F5CD95EEFD10BEF5FE9A4EE310716C4E98085"
01776     "01BD584CA7C0EE23976D565A9499210A95A3D2A78D7182F490098927 16972323ACF295881D"
01777     "DF44CAC78983A4073D8CAB2D0442C15158CA46D55ED2BA8950F2D85353CFBF6525560B5743"
01778     "F61833101F37FFED90F1AE204811952000E608356105E43EED3CF09499A8593120BFB88FE0"
01779     "ECC4AB465B9F3FD4065A2A1951B872C92092A5ABE984E5B24AA606E711075AE6CE179B951E"
01780     "5B45B54EF72F8787A82DC82585A82D2A0CF368286AF117B399766A8A4DFDEA87072F18FC0A"
01781     "847B8D485FD28AC3C0F58B3DFEDE3E076D56F27DFD19D244C7564688EAB0197A61CEDFCF15"
01782     "98D8818BED9A53F973DD50CA6DF6FF651D3B8242655F0F0C9D22A886DD804C799AF860B380"
01783     "2C270B14A1FFDFAE3F92E09B31C12574DEB4ADC8684CC5B47F87C2D905DB1BADBFE78F1B73"
01784     "82C35E14BCC2AFA0020B942E1EC369E99CE9072C4CC0155ED1665839EEA138161EA1239AA1"
01785     "0A5BE5A6526A5AD6905F5C7EAE5866085F64A35EA9A8ED52E382DF5E42D0576008BD378A5F"
01786     "B0A289C02EFFF02D6797110A7777D54845696BE0A7A78BEA81742F2F98003C55129A47D1D6"
01787     "2E712A6B8C04B499295E0957A5F304A91AD0BC007024A0734EC99B769CB0C725D875B08C35"
01788     "2F14C4A6245312278F67EFB74E16461598CF1970E30456C1C667E29885B3B291340514924D"
01789     "9F726B8A3EFF64CA0248DD03C45E5363FF1F17FB1D98E51DCB74848A028C42E864A3E477CF"
01790     "90BA6D14BDCD58250E849CE88E19DEF46476B726DF3A4D4A859BBEC49AF7D1B4F4F2A3101A"
01791     "FB0D0BCDB082C712C20C7C8DCD344361E3D9698C3A7FCA20BD554B954ADA52AE0710549CE5"
01792     "7B5200D34795C91129F33DCD7DFF9F3EC88E72D2EECF48376A991E08C682C4A18EE1D97A6A"
01793     "8087099882F775E62ABB5A75408C505D9DCE34EFC173EBCA5C14DDAFFCAC5CF77464A22FE6"
```

```
01794     "BA0771E13C398B7AE6AAF10CFFC8CFE238FA9066F449AF172933BD0AA73AE90E406431BCCF"
01795     "4BDA4598818775D4D6233CF1E696810552890BD793B104F17BB1C219F52F6028EC1968F9C5"
01796     "32CC723DB3D18D261721CD922443ECE10DEAA1A4703DA1393828DAEEDB4CE3541382AFC79B"
01797     "A928E7255920A3577D56EE8192F43B9BB2DDA58294B74C7C859B39EA7426C3397208703DA4"
01798     "49A941ABFBBA8C0DD790B6FFBB1AC5F5237B59AD1C259D68E1775C6D001786090F3DF27F73"
01799     "D70A11C2491A461B7E8E58584367 18C254DA1511221D0A6C8CB88E76D3243E3815857B556E"
01800     "CC9F9AA3B198896153B287F80E68B8DA221E14D620DDBFB2B76DCAD209C6B13876CB044A4E"
01801     "E78FC01F00F915F62ED2355EE9094071AA99BCA1F58A9F1DECEDACDC4FB94C26DC54C447A0"
01802     "8FB4CFDBF6E5861F83B3E9277FB5DE96FC6C46C3CEF41ED9F187750FD51597D5A3D1580CBB"
01803     "3DD28DEBA103E036FC8B2508BD1A6C558849AED46C6EC216F6BB9306CEA01FE33E0B82A5AE"
01804     "7D18DA377FB5CCE75E6E3022D15D9EB3FFD02D22E2530D3D342C561F59BD0989865EEA7F59"
01805     "FC5F197D769CD6C62738646810ED1C94D87B0D47C13124572DED11BEC718C8161C78F0DE88"
01806     "0A9E39257B2DE14241A0A5163F0B3E41491EDAC323A050F7D75DB77821608C341F5DFC461F"
01807     "84E7DA25EAF86E7A7DD99C1A004408DC9DADB36F08"  // PRN 50
01808 };
01809
01810 // Galileo E5b-Q primary codes
01811 constexpr size_t GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH = 2558;
01812 constexpr char GALILEO_E5B_Q_PRIMARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][2559] = {
01813     "E49AF0472DC849AFD1529E7227F001FC9FCDD82A502640F54EF88A2B201F911A1F90562966"
01814     "D0E4101A8846F50D25B0ABAF49171E3A0C8384465B39458FEC7D8644CD6ADDA0DE47617495"
01815     "DDD1886FFD3A960A255D58DAF9333F73AE833D840ABD6DD1D749C20B8EEC0C29E63661B0E0"
01816     "BC50E9EBF4A4CDD0FB5B48E141C16E196ADF48314E28E87986FA609DC9F11307B68EEDFE07"
01817     "7AD6A5F8E7A333DBDFB74C126FDEE4E13E4C879CB19A381067DF74A075176104A3C60FA17C"
01818     "934F01CD20986A3844CC3395C2F14DBF38B4932660051732A813D7D224D6A9A46347702B34"
01819     "2A8A1F214550A3E30DE0D007029BF701A32DE08148C67E341C95FDAB5D31FC4AC790DEF152"
01820     "1BED12156D12E68349B60431261D99176797A9CD9F855D3746D03C575D6CED7E62C401E24A"
01821     "A204A3571AD6316A9F288E02E99C4B29AF9FCAAAEDBE83306694FF697B2A894E350A3C2DA4"
01822     "0EF7CB537E8C046368E47ADD96BFE357558719 27A94AD9B7C5C70773FA1B707FBAC5483946"
01823     "B33887C82C27F7D86045749BA8717002FCAA21DCDDCFCA7F90CDC9AAA43825374066C254C6"
01824     "5D7EADCD1BF4B2F32E0ED7C4C93A62F00B652938FAC680D874461D9F8624F23E26F7C40F52"
01825     "AD7668AF00CEBB7AAD8A47F144377BB81D9831DFA53DA0AC1D6C1B909FF9708B05A248EDFF"
01826     "820A84B08F94DEC6B1308C2DC6A44473CAA01A4CD4124CD99130899FC85F7D2C7FB1A12E5A"
01827     "93F75DC461F00290627EF320E3228338FAC94318B0D767956BA42A38B58571BEFDADD77921"
01828     "326E5C98F8983BFC70B0CDDFBD8C628A1A5CD6BD015767691E85A3D6FC1702C5FE9ED26DAB"
01829     "3B8148E28DA4AE64AF460EAF272E262BCCB5F47209438ACFC0743BE66C9285404C1703331A"
01830     "6420CF3BFEAE100F555C633F91FED3C382E7359ACC381A1E1044148A2BFAFC132833E74811"
01831     "9EE745A927D24B72C1D9296AB2A28E6ED23CA85905AD923757E60528B78804E37CFCFEDDC5"
01832     "5F799946622E9F20BEE58193FB78BF63F53116ADA238C24F44ED12EE018AF62CC20CA45B1E"
01833     "E8D6344D35574F395B065C4BB76074704334B317A512FB60E24BC5B3719C471E0DB67010F6"
01834     "22D4B65400728CB7DFB821D63A79DC10090EB8C5D7E56F939F9E5339C29EE1673ED994F19F"
01835     "72FED83A247FCA65E12F3417B30EDA9C9B917BCBD372D1CE32520998E60E54F1D54ECBD87F"
01836     "4A5598600746D30011FCA3764686972C7FCD3CA7E5DCF79E226DC61C66AE6C70F2ADD41D34"
01837     "407F847582110810BF5F92978A50D5AACBE93F9A3632BEFA257B972D0996A6CAE12F87B8BD"
01838     "787016093BF64887367B84559C215B2C148403102DFE1D143F6724DF7078CA2EB855FCA5BF"
01839     "334010810EAA5C03094F2CC9EAB888BBC88886D7B1D83AB723D32FD7B6F80F2565FD7E087A"
01840     "4F6C48354C0FC63500DD7DCABAC255BD92AF6A6918B3159EB776EC62E6B4AF9A7A397C75E6"
01841     "963B92515FC9CF1123577B069B07CFE5F01E986F28A6E0883793F79F41822944AE6CEBCACB"
01842     "4CF4E2534CC70EF682CD066F4EE142A55844C104E1239F51D69D3C6BA1ECF2720243C340B4"
01843     "E04DDF68FBE8D5A07B26F4C97583422B3C3143C2316BEA8C96EA365055AFB39B719DFC45EB"
01844     "B2A52F52CC67341DE493D5A788E89671B705A6DE2F92FB0ACC6C31E200EB38E689F64D24E9"
01845     "02C8AE95CC75B22BF9F44DC95130F9CE91B4F1C7EFA96D32001DCACC26D9D3F55B0C653D8B"
01846     "09AFC1F7DAF71C721D55F82F89E3DDE413080E8334898A5BECBB4B6FB2CFAA0D2390753F08"
01847     "62BA6C70C6CE011783C6240712939C93DEC36235A8",  // PRN 01
01848     "CE701F7B81A1EE81DEF8354D759C27FF6E4C6EBED2153250472ACB70A82E3BE39DF9DF9BEF"
01849     "543B6645CC6C5595565710BEFBDCAB8654CC62A8B60B6A90492EAF1216B47227F0B088A43D"
01850     "4E3B5E81946A7F49A14BF917765460C80543EE8867C116970B2957EDD5F7110654AF69375D"
01851     "673C72580640146C7A4F640F6524D3CD827342BFB29E5A211FC373C94E11F16771D58D008E"
01852     "DF116230ABD8F165797E8B0CFC589FEA57788516502A664C04AB9F15310C55758CADCD63FB"
01853     "E743FF9DDE95AD39DF253A669B1B5286C32F379AC7CB91973AAB3FA7759255DB068D225FD8"
01854     "5DD5D8AB38484F4D09ED5AB3CDE5924851848C695EBEE8717B03919256AF0F0463CCF030ED"
01855     "39C5B38A5F4E5CAC680389FD80A7FB83868D09D18EA4BA07B19312BFBCD33715B8CCF23800"
01856     "60BDC7850176AEDB634F21BC15AE9BB76A879C76F10AD8A09D4AA8AD2E4AA1186D744E2A15"
01857     "B0A107C3921C10B007A0D41FC89025A9CCAE862BE52E136611F6F0B3742E115F29118E09CA"
01858     "13B078775CB26DA2C693BBEA490F3BD7882D397DC69F50999EA9F4BC3C276F6951B0E2281F"
01859     "A3CEDACFE572930350BA83D0055D492C310BD8E3B74582BD98CEC33847266015E207DE50CF"
01860     "DC1CF01530E76185991C238E63CC4AE2E14DAA55851C405EF1F38B7449C1C626555403069B"
01861     "DBF7D5CA63355EFC82F85B3CAD343BCF5B386E6974DF401E130BBD16A44F282F7E354048DB"
01862     "322D9CE029718DD62574CFE8614CC891AFF3905968FF8BBF0D6FFCAD3F738FA3B6F4871400"
01863     "7024DC10FF6430FE2B4C9BE4EA2CDC2357AC71529F4362D9A792D4472E14682EE124892B8F"
01864     "186F397810496C4EE7115E14D991D51E8B4FCDED792E53CEF163726760741FCAEF963E4169"
01865     "B20F37D5B9627F62A503CFDA123AE58A780875171854D4D2C299E7B0E8377CF040313090AD"
01866     "9C53B4C79C7E49BA110A592673122871CADF826AB15B3E97275AC1CFE3948EA832E2D0EBBD"
01867     "317F8547CF916453D57A7A18D39CD3295B25977FDD73949B1E752CBCEE0C285DDCF38FC515"
01868     "E509FFA2721880892BC6B911289C3D437BE6F58E36FFEF9557A520C772E620597C691E85C6"
01869     "A63212CE7D1641A41055681A83CBD1F61024C6C475DEA3598819C7BC33F798D0E0AF481568"
01870     "52F9E262835607D496AE6A4C1261ED97D3BDC8D63A93EDB484089E228358AE37CB5EED7EF4"
01871     "BC2825F6DE89F79E4F8E5C8483479A3A63B0B0A5D54F5F79AF2A8657E167D08DC73CF1D912"
01872     "85DDA0B1277477B4E4B348F4167443B295730F4BAFB61DADC7F45D4EE9938D195F76CF1FE8"
01873     "947CAA68A6D59F2469DF27FA3DD5068E7271C55B42F882B54F5840E5A4FB09DAEA73ADE905"
01874     "604018C722CF9E1751C6998275BB40E8BB254A8948C13997BF83DE5E4640E8D40300FE6F70"
01875     "8BE7B1CCC62063E24B4DFA3550CDE01EC28FFE7FC55F2732080E3FD4F9D06D0D826D49A130"
01876     "2AEFF95D88C7ACEE2E13DA40F29175030639DEEE713F5A3BBE9488349E594A5B2A27730B18"
01877     "C33943A89552D17F5CE967B05C55CFEA5E23230F49862375B8C6196650BB9B1B39C57E1878"
01878     "A67EB2C59D486CCB2D0AED2ADC102B5D0FD88C62CDF3C45221A76C09FA43F2924497B328E3"
01879     "AE725D108EA1907FDA8C8F75C1F53225DFE87C3E136BED8DDB491D0BE615598C6D6DC0C6D3"
01880     "3E844E3DDBFDA299B1E12C11819FA83F9BE494A081862FAE5F687485E2264D9D7F0117CA3F"
```

```
01881     "E00B0EF59E03253BB2C5989B770951F16DD8B15E9D485E163AC60EBA10974F158D0F121C92"
01882     "74B15FBFD3C4B639F04D32F3A03CC2E6E32A7B4AA8",  // PRN 02
01883     "54B709AE2BBF27FCB2730AE75AC1B3CF690A17A43038405FB0B8F87E2C9A2242050998FD7E"
01884     "79B96CE5AC5CC9B986632BE6DF55BBB7BC0483EF2F1B27593C0408487E8F773A75FC51DEEF"
01885     "FF6581423D0155440563B04767549F4965089102A0ABE280DE00E9F92804D386E2B84BD9AF"
01886     "5D7C1F0565B75F04BECB19C156776E75C73A4855A5E18DDF796B8C8E7DFFFCDC51F3AFE9BE"
01887     "FE8DB161FFA8D6CB37044EF3E1906667BAC68EE3CC9A1ECACD94C2FD20B6D8EC553796DD9C"
01888     "BF3A4F81B395F7C2C8A56CF8B6CC27069A352DF4CE53A783F94823C233E4CB0E928335C744"
01889     "B75649E2518B8A8BC6E09A704DD025B73FE9FF962FDBF4FCF25653026CE0E66EB947A33727"
01890     "643568447C91B9DFA9623204FD4F127A708311CD25F7458E97F408F76FE14AB7267A4FE4AE"
01891     "E0D28B6DEF4D4309C55FBD6886E1372EC8FE61DFC89DEEE121898DE2EE7EE661F436BED885"
01892     "0E8C28F569D2CB2C9944D4F88384C0907A8076002E5A3E80F09FF52517932D78A647455B3D"
01893     "5F1AA72F9F70697617ED745DFBD402331BE6F72238C7E8B83C6A7B0DBC617D1C4D414AC016"
01894     "D337ED5043C0B7BBD46237949DC9E583C580A578FB36062F5BC5107E5D15508188FB8FA983"
01895     "A01C556F472C489173A1A219EFFFEE9815265EC1F0E87ED545BF55CC80AC26F5A17A04EA1C"
01896     "8167ADB4E97194D515B23B3DA5F1B77719B37F5FDDACA0671965C8AF2F900B0F1A26CA56EA"
01897     "0C3A801E05AE3DDAC42C5E975E7D8D65151AE2499702CA49B905F8BFBDA83225E363680E76"
01898     "B3C84442006EB644B19B01698BB63DEF284DD9C6B78E500EFB4C10AFFE3F5B19B755A9C47A"
01899     "6394551BE0C01F56DD60A7A0E6EAE7A57A74096142CC055F5883BE9B21010B7D6B1BF525C8"
01900     "0C25DAF89E12A8C9BC8B09B7A796319D5562B0942421E6BFEC5F9A891C28EFF59BE7D93BE6"
01901     "A9AE3CE6BADF0057B009C3C83B5A70A546984181EB121D23ADD7DC74BF93118AE200344E20"
01902     "0DA0496A930215EF923DF384C69E222B4D6C95D3E6698171205466B042E1EDB58F786172DC"
01903     "DD960C6F76992ED7DD5B8D88C8D4A3C452D30782206032C4D50AFAD089531863F5D3D12D7C"
01904     "12790E6D68C5AD406317A97CFA2970A0F5EAB7A6AB48F6B246BE7A9F1C17BCE38286190CAD"
01905     "21C0C0A2EDEA69CD82C0B50BD9CC9BBB321D1AD292632EB0D3793D6DABA3CA1217B25C7296"
01906     "4464DA56A6731F1CD9868E4678122B0FD38DC51418BC199BDF6185FC9BC6C4596A6383A540"
01907     "9D9C32EE8415C3C9AEE2DD158F7482232BDD80A0D4E78F8E452822D360BA6E34AED9E50D44"
01908     "D34702497837AD5E095A4DF9ED4C018DF813D5D89E04BFE1E62424AB17E9640C6F7503DC9C"
01909     "03A85BC094B3D3077F263E11CA128EB8C14A4BD4F26DB8FFB2D0315B38EC76B46121E95AD0"
01910     "3ACACB851DD792F8977056DF573E6F2B200FB02B9C26148EB88EFA66514C03ECD57801277D"
01911     "77744D6D77C2AE630DB858A29DD219DA84A814276F7A03BD0C4873CBD12C883353130DA044"
01912     "B887DCB397750C94AC5E047683D457E25A3E25E34D7C6E8D5905E2B04153ABB905CBAFD341"
01913     "3ECCC4C022754C05C0C0814EEE8284FB399EF2BA34020851E1B233C1F4095302F2FCB07BDA"
01914     "74F1483B213E74FC9D1A1B0FB1FC00ED8B59CA2EF39EC3099713C704D7D4CFF0D83E95B1B5"
01915     "2BCB0A6B5AD81A9EC988E8165C8C1BF36FE3CDD1F416FAEC8E8255ACC758194064F4F75BFA"
01916     "AD22F2D000DEF719C09F10297D2D5307ADD524E23D44C36F1672D1482D15399450F46EF33A"
01917     "36466F34B15795388CEF2FE5C060F32F337AABDBE8",  // PRN 03
01918     "641AB1974308060B0EDC824D9A911F7F95EE18C74C95D97E6998D1D8F2137FC6B95615004C"
01919     "2A87747D65C96A11E58CAAFAB597FA9015888E2F66E0AD827DD2F3CBD3322E9D9B700C67406"
01920     "F154C3749C349958CD199D7759C0236AC94868B15DCB0C85230CAA19D56C7BAF7A761C61DB"
01921     "809FA99E63B394DDAF765643535339FC270EF76F4734BCAB1938397014D334221E05CD5C47"
01922     "55BAA521BC6933120825875F90FD3CC65651582B7F60BC3F14D7C289472776CE821769A4BF"
01923     "8FD204C0298C78C6BECC7E209666210593F7E0F78DE625CB684B3D4EBDF0B29F965FA295EE"
01924     "C2F0CA520053DE46514DB165A02C1F218D4925D4AB4EF5ECFDAE4506DB184A181606DE65BC"
01925     "857AB778549AD8AD878D11A0FB9AC44504359D85622BA6BBAC2F8CA3FF4FEBC0DB05A98E21"
01926     "14A1B58D34FF416BEBCCBF714BD42269CA84853864C21F3DAE1153390EC1BFAAF3BED24B70"
01927     "D00D335DB04F528922A2A17F188506407F2FC7782B33DC5D87C7F0B4B635CDE1252609650C"
01928     "994EC0467229299766E64BAF5C2C094AF27BA9BD2E4CBD0BB67158DEB6B5E8A0A1C8992554A"
01929     "FA3449AED56D1F0EE41E924B907507BE3BD7996EE08991F1D55A8F1EBC949A34B7031DCFF2"
01930     "049166489D05A3FAAE6B4A1F1B31607BABFB74A0F92537E1BF76CFEB1F76B56CD21E1F262E"
01931     "D59BAEF09EE800FA1A9B30DF7702D8CBF40AB6B19DA2D4D6573653CDC092B7B858F8A4407A"
01932     "555CBBCDC7D6C8B671032A5DF38801593557AF5E7BF9B1640A9BB183014DCDC068E172224E"
01933     "48D04023273F795E3F1B966618CFC7840A06518B3CCA1987B186EFE749E6EFDE8F1C85F4C7"
01934     "57F5F628E986B093C2F46060ECBD40D9C8E547CA9AD57D9C62C411DD9E484C2228F34E6105"
01935     "E7A94C26932DEA29C2437C1EA011D93919C029E08221F633B4941EA1AD3B8562A5E0499D01"
01936     "F236811488D9416F11EAD0294A959268BD988F946205EED0B98897DA28B21E75361F5F0880"
01937     "4C8864D14B674C91639A9FBA0F1B0DDBE5073C7E9FC157688E48A14E5868AF395C5514BF30"
01938     "931B0EDC45EC49C5D3A4DA377E129CCCC5E284253FFB7133662145C3351BDBDDD0BACBB370"
01939     "DDBD193C23BDE59CD55DB29E54FBA2D7F890063E272E2DE88BFF0849B04F8F24D1A0CBE0CA"
01940     "F25F815BDDB7D4A318740654014E89D12683C0C2ABCDCA3456F475A9D79D7FB6858914DA47"
01941     "9FD11A2F545E113293DF5D8B84D0FEC6E49601606025D8CBF071A4980BEC07E8CB3D204754"
01942     "3D56FC32DA90FB77459E4E9C7048526B31386FC907AE118ACCC050403F9AAF5D530CFD27B9"
01943     "7238B91AFB9F36F2F3D1E6BD5C8630D0C6DD92ECB212D87FCC314AF7F0FF49E50A2BBF36E7"
01944     "D19AA5C2D785F1926EBA6922583E885307ABDA921F99C44565B2FF4135CC3C3B6D11AC2DCA"
01945     "0E3844335F8A5C3387B6795B98C468A48E87994B110A3B3BE39DA16586171F33E1162623CD"
01946     "4DC3E77C3A887B942A7D792684F7E3EC9FEFC08276241A2D0051682139D23F864A31358D83"
01947     "415F4E2DA5AA16660D74C00539564A0C791983EB1BDC5384C975DA16605B9676C61EE54487"
01948     "7A27F5E6FF1D1C6F239917706AB2FF9CFD4FF0543E7B33AB7D573467041C88A83AEA30AEE4"
01949     "84A6A2F2C73856EDEE40101D66589D52F44867ED29506376AFE0992CD3318A674B055EB9B9"
01950     "70E183ADB4CD6BCA41EC3DCFE7CF14033FCB1FEB4C38DC18B84FCD85DBF488A4316C9C7635"
01951     "12D6058C2659645E2A2AD79294DCE325D00B9BE94B78AE970FF25109FD7BF9DBF42BE0EF05"
01952     "8B2771B7BB7455D5905CB66EE700A8DCE17645074C",  // PRN 04
01953     "FBD0AE56FBE835762FE2E970D976200CE1798C3E9C416B2974967BF06D9C937FF318888EA7"
01954     "D2763BDE785CAA9054DA3315793EC4C53815E0397CC274D510A81DCEED4B3D5232D8F19043"
01955     "E567253F0B717D6E7CA009A01592BF499D4BEA95612D126D46D1B62E2E2100EE640D55398C"
01956     "AC05EAEA7045AB515A8189C50940E15F2DD574F9CC46A3D0D3C0B3DC11D9FF2DD0453B6B8D"
01957     "8C28B8B3BEF35EFA9DFDCC9A6822EFE365B0D1A7E63A86B92A2FD34FEB0364B41AD17830AE"
01958     "9025EAD29AC32C1C6722420F5F1C8F9570E3C39D9EBAE407EFE9EE4D54BB70A5525039AA38"
01959     "CC109582C46DD190DBCF28E94557986C53724C64CB02487DE67ABD26895D3286154D2DCB91"
01960     "D66DF17D6FC53E3F4B02F55F9DB19BDFB744D9815E35FB285E331664E99A41DE1278C5051E"
01961     "BE8EE81904F80712F12E92034423B05AC4003121C55AEF7AC8807DEBB0461DF9A6F92027F4B"
01962     "8829BA4D81A79A68653BB132D1F02CE72547FC7D36B69084FD1541EA69F6648A0F2BAD4D45"
01963     "C95FC0B9F34962E8E4A3D001D515BD1133476810647DC79989E7B7AB0603E9FD57E156D137"
01964     "34071A136D2856040E09ED9C6E13AF417CE477635BA9B7D33C6AAE826FC5CE4490AD53B537"
01965     "FB8CAEB110E81DF15BF7FA679DF85F2643EACE2D60FCADBA01119792AF6AD86C88E508A678"
01966     "4BB679851EB11AE07492A8E963C13BD7914CC02F4836DAC1CFAAB94E51DBDE7B9D3E6B4DBA"
01967     "96DC8E7A38B8A5C58E724F417E01549BA56DE0B711233A370D51EC73DA83EE1F8D92B343F5"
```

```
01968     "B1ADA8CADEC9774623B0C59991E9A09E42D563F69E54AF3A6D286F88050E01E562CE23E45B"
01969     "8366B9847024C96E6BCDF756AD0183FC8725738DE34C3F6C9054AFF0B17EB7ECF36FD08505"
01970     "00E258C3AA9BC4D9E988083346F0B03C8FA394DCA5AEA0F8E663520B726346A0A13498E3CE"
01971     "45845A2D6C4A8AA14E1488916B3751E70C18C98617191F785181E7231F2DB8C712A7DBD034"
01972     "0810F07DE493C7ECF8D6EC3E9DD80608A31D0F4E0BABAB397704DD4EFD12D8F27541FE02B1"
01973     "04973D783F18223147D12E09F5A6D0127D2BE9C5EC0F5F348AA61F534D6551991A935CF910"
01974     "DCB5DCB09BFF6E8DE3F4C17E2DF1CCA2411BD91D074A3E6295CE365EACB1A3FB61AC2AB85B"
01975     "8D60B01E290D3DD1088BFB15F7FAE3D534011B7F2D985DDB4C82B6454FCF80BC03338E0DDD"
01976     "12B080CE9821F09A07F790B9538C20020C462A535773A9EB1F84DF63678FF761B8251A36AD"
01977     "896987006161C0B3C9A5ACCAE1E1B2A7CD791ADDB02A9C41A861EE6AD2B7882D84A7E4AF7F"
01978     "1F8D15CEECB25C3770ED938F31A8F6ADFF298D7FB7C90E96677A7F0A78DE6F4443C61E357D"
01979     "3FD2ED9A0309B65185E02001ECB4054EBCF7DD0D3D4926A1C02FC39EF9E160A81CADEB6A43"
01980     "922D46C25D9C15312B8D634E72FA49EE822B748EF959B0D5545499FA75DA1BB24F48D0BEF9"
01981     "D423AFE2BDA14E66266BE6FBD5754C36881475BA4CADA0E2A8EBDC6A25712A55A6D546CF42"
01982     "2726AC8CF34FD8176FAA5650D298EDB15A3663CF28BE7AD5D0B0A92F429BF8CAA807F12A58"
01983     "3DEC5890D3FCBCBD2731E7B36EE58E3FB2670B301BA7DF1A7135CC2BFBBCF052D268A6A636"
01984     "465DF743F2886B985F8DE35B8428D6515F32C6F9FDFD3C04D3B2816D416E2906FF7814E82C"
01985     "9178836C2573C545AA976C5DC44DC87CA242360861C97332AAE38ABE73D556460316FCF35F"
01986     "11069C8B25D3064EA9F62BAFC0759F419EB889849DFDA8393A6B32E1BE7C4FA7B93BCE0FA2"
01987     "1898D879EB0E3AD4CDE61AEF8F0DFB33E408E9992C",  // PRN 05
01988     "0D8BC96A69914CBDD029FE74ED2E51FAD1714E703FDF5E3475B864D69E68453B640813982F"
01989     "4164AE4B86CDC5233D544387215FBEADFB53862BF19DFEEA39279DDA4205C4371045A62071"
01990     "14E151B055FC03FCF21E8E71631B249E8466568F1FA8D290D63AE42121F04877213FBC758C"
01991     "8D1D5B7D7814C9778FAFF5CA280D2FE81CAE909DC0C4D29315E7713F8E1C64DBD1D91079A2"
01992     "6BCE900D29AE70B8820264A4BEAFAF6135EBA6172C887B5A64F7AEDCE15C85E4CD8222C3FF"
01993     "3C16B6EB153F190EE23BE455B48757086AE6A5FB7474C01A11469120D8C5618B315777950E"
01994     "4BFFF3D5BA4AA9015CB3623B706B81F3DB5CEEF23B93F111724D787B2CF9DC9CE02E1CD35C"
01995     "60D86BFF8A68668F2C8C531238695478 89C72BE1E54F261D3755A9D5EDBE95F72FCCE57643"
01996     "7787E3039570FF6F9A67B675D57213EDAF80C2C59F46B3EE7B27E54281CA4DF4581AEA8E78"
01997     "0A89E46DC36EBA3DE258F38F963405A6B0DA44C03CC7ED52043252141056F4BFC4F57EEBD6"
01998     "F556D369166FDCA966702BABE5404653CC31CF17354A9E6C9EB13EC892615380CB6F3E891B"
01999     "FB3B4E39B191982855DB272117C758E5191873E537EC9DF004C4C9A637C819C9538FDE6EDF"
02000     "02F900748CDC07B1CEE9E20717166B4FA6F5A2A9DB613B9B8D4E9E0D5B65C73198B7C9BC20"
02001     "14DCCFA3A8B6B4564D2CC0352DC85BA2FE18E2B5DD9D348E45E9B17C1A2FB24D1297217F06"
02002     "A5FC96480F6D7438E1187D050E5ADB681448DCD2AF16DF7558D90B06DEB83259C2DEA3EEB5"
02003     "CBBE882AAF853556582F423304C62C1794B72E4C4EF47AFCDE04128065F8CE6CC0CBFFF0A9"
02004     "F493804E7F692945DC98677BACABFE1C28F0E041EF081D8A24428A5FF845C912D034573098"
02005     "401C292516CBCD1362A72E44D0E0A88ADAB019008F4C09E82FC8E17DF47CA600F090ED9392"
02006     "086C37B65C18B3F86DDC23A1BD6CCE98E5F671CB183F90AD3A0E90F0FC4DFC3CAFD9CD3C4C"
02007     "D3E0C37CE41067317849A58181E079726120095A4C2A71001037CB003DA53F566C9C474F3D"
02008     "82C5688CF9D0EDEA58EF17898DA47BEB7CE2BDF7FAC23F5EED9D4F31E810A2658AFBAE2B9F"
02009     "92B4FAE99169584FFFEA64AEA47A97E3A723EBD291E054CC790C9FB62661874C61432780DB"
02010     "76C2C667B78DD6FF2639E1BDADC7A7F010AED1E48EE6E1C9742B137ACB7C5B06477DC55CCB"
02011     "53F400904020422D93F60316761ABBFC4BF664836CCCC676EA8C818576E3A505D80A8E750F"
02012     "456201395B6BDAA1E180513EE17AD80EE3A06F8B36CD3F9A6E041DEA6362B11FC89193454E"
02013     "BAF0A01E1918BABD7616D9366B6A33E890F9CD237E43A551766DCEC80FAF344810AABB9651"
02014     "3EFB30A72C436C22959C7D9D9A9F26F5B24D79CA7921B75A9DD8DA4B068089EC5AA09619C0"
02015     "C2F591AF17B1A16D796D61A7BA98AEAA12F9F444CFC695C7AD6C826B7E82BB599925193390"
02016     "08EE065F485153FBA0C6326F58C2092AFDDE5BEF1D111FE65AE008EE249EA4CDFBAE7BF084"
02017     "A59E38D81FC0C4AB7216C84C9104316439FF387909CA66A7D88520C0F2137B5465598B9386"
02018     "9DEF41DE46347B20146A1C274E6E908D2CAD97517EC7732EEF179F69C299F227B2B8B385DF"
02019     "F5DCB6103C95A4A3C9BCE35C46330AEF530541953B77C2BCE29C1A2974C6DE07E0F3DB4754"
02020     "3EEF21C307274567532B150A93342D1FEF7B4675349089A1CBD395327E61F2579883D4BDCD"
02021     "1BDADD06721F1502E46267893BC76560CBBC4A1BA02525AF67C6D023C14B38CE284333253B"
02022     "522E068C97374D6B83635E70D9BA88796342F5B998",  // PRN 06
02023     "805FA5A4A543914EEAEA33F40E46532D09CD4CC4CEB1E5987770B405EE24848BC2170B7B05"
02024     "72D058E726BD79E7DA005F0D6A11A083CD2BE82DED9C5355C254F645B98EECD37C0F0834F0"
02025     "C69E1AE681A69BC2A0F108AF1205DBB51A6D5B50D44795DF198A94E1C4AAD78B680C84A7CA"
02026     "148EB6698DA12792FBAC8FAD6F7CF086899E147B645EAFF7D8F4BED2C7D0DD69DC3F0953DC"
02027     "6A6CABC1E3B9F72C07F30DDD0971D9BDD11C861016D0DC73A946D95DBF07248BFA896AEF4C"
02028     "21FF57D76F894399585EB2AF7299472ECC6D5386C9E219D535D8CF522737A44A3C416BE882"
02029     "49CF607D5DC7EE411734AB878D5E10D61F7A2B501F5754740DE92397F69A40B8763642496C"
02030     "B141EF5CDD8AE9454F6C5639EBD70D302E92C7ED592EBB3EE0F6F6714E113403726E838385"
02031     "3CE8E59C3400C00BA0B335B6212035BCE33A91C597009F8BB933BDCB3481D2F790499923B4"
02032     "5470FD6EE3F2EACC6BA84B686752F826D7D0804BC53E677DA1E5198497A91EB071562ECB42"
02033     "E38B3F5BBAD0A47AD06F4F6972B8126618E773C35E464962FD4E338C33273A167189C8AE5D"
02034     "7A62D78302189F57FE04FC420843137DD8CBFC2896174465DB4D178256170B7DDEC0D8093D"
02035     "0951A5C8A8B25C94C318B56E7E2DEDF2FE39E49AB6CC5D3B3425C82142866EA2F1C1FA55EF"
02036     "B420578D7601F90C14832C7424FD51ED9310FE62AFDF7648E18B8F6340FDFB50B6C34992DC"
02037     "45C06F5E887A511408FE836464BE40E5F672E09E4FBEC8FD486B4313BF60E2F6932FA9BED0"
02038     "ACD0A76833DA40E811EBBDAD6CE221656303C7E881F63E08E74DBD10054C30864CDE812016"
02039     "293A419A32BF5400093A3F9EB405617B0B6AF64ADCC70880936CF9D4EEB17BBD8E416AD981"
02040     "3F1D08BE078145DD493EE752D3954BFA12136FFAB111625948C5E7EDD13B2018D22B6089EB"
02041     "4FB71698BB7D3B9B7A1E445CBEAB19DCF146375D0FE53D24978F7B5D7222660BBC5B952117"
02042     "06E865D98EEBF4D3E88BA9137CFB74D18B7285F540E2770C9DC2BF7F667FB3C7708AA4B3D1"
02043     "CB8A71E374A9BD7F3AD6CAC883536A78AFE696A7E446F766F1CE5FB5F6F7438F2FB04D09EA"
02044     "0A27945DB95C29E334F526B699C9758B2654EEEE0C1D1ABEFADFE7BCCBB4E3472129B8F8F0"
02045     "83A63564A6856F5223B08A6FE3181878C64F3C83DED97DB0DB3F110869B7929BA2F9C3715E"
02046     "1717779AF88A061FF91ADEB7F1E293E822EB38947FB6DD1026980FD33E118FC9ED9C69AAF5"
02047     "3ECC127A92C82FFB025DA5C6396BE6E1BC25D4950031335A2E24F5DD034563700F9F8C9BFE"
02048     "35B1250FF96E34E4BAD34A864693DAEEE7979E405FD7302C47DFFD3EC860BA280EAA116811"
02049     "4CAE3C229F1B76BB5B486534C7E3D9D1A871CD69AAA1FECCDF7D57FB8187209878C240C9B9"
02050     "4CF9224ABD2B0AD649C71B850E91836A293ABA64F12E0859CACCB8A0F52E3DF38FAA483AD9"
02051     "231CA0B797FE57758AAF7D8E7434E7BBEE4C8FAA8D403FF1ECE06443C19EC02632CBDD0F15"
02052     "498D8CD6F01B23D6300171C8CA10A87C321A2891BAC887299607A644781A0429657F5139CD"
02053     "2BB2147FFFDC8DD18D16FC0B3B20DC1E21306DDD17981E5AD27D26CB43272089E40802D484"
02054     "01CCA1225E3F55678C5093C19A65C9FB882C295E3B3DB5F3A94EACB5B0D9C960FD7913A997"
```

```
02055     "904610121D033BB02EED60754B4E7C0E6C241A95E0A6BC7A9958F8256AD9F2EDEA9B6D6CE9"
02056     "2F7AEF8BD5D651BCAA6483CBEDEF5F9550CE2941D7164FC01ABAD744A0E04DEB0B2EE26226"
02057     "20A593D8F8188C5701967F70A2592E6E3D8DBA4838",  // PRN 07
02058     "D86BA0EC5F2F711EBE19B487C0A3DF6D083DCA33E132E879AC4116BA4028CF2D131A6146C5"
02059     "3532AB0CF8969C0F71C8CA2D62AFCF9D98FE3654B54F1D09DD01A03B1D56A0D49F0C651DF4"
02060     "A07B50F525242CD2C3A4FBABC116BEA613D3B7EE5C12D4EFFE45E2C2A13DBCF3D893497A10"
02061     "70D089232BF38A2546F381F080AF51D7AE4BB9A7B6FF7D70D0D9FD49D91E621E8BBEAF4B06"
02062     "929A267922923B9A31C952B83B60D6C667C5B177F3196F0AAB6E8081F6CC5F222BD17FF2B5"
02063     "4C9BE6E51DAFEA96999A7321C458BA3D87E2DC1651159CA8784B416F003B525CEE296EFD41"
02064     "2973D8BBB41084C9215C49D95ADB95D8F763D7F1E34CDA773A3B3071454E7ECB81010B3023"
02065     "2C4CA479AC5A03E7C694A8F27F73E4E8C02D48F9A31F6262E8EDE318D0EDFE90973C5A53A9"
02066     "CDC46F5318D8B21C9D55A8FE997DB6C2A76BD2FBF3B9EBD5FA58A65279F517DE527CD91B96"
02067     "17BE9C8065AF3B0F141C9CDC76B54CB352A249EFF27F581E592A46F96FCCBF8DFB6683737B"
02068     "8B3C1EAC067F2CD0103FE3D98E5C1D05644AEAE2876660C423C76E088F78B7EB43DAF6B6C5"
02069     "B3F7742B61F71E4B94C6BD65A1CF726499DA0356773C709096E0896194EB9CD737CEE94B0B"
02070     "A86469D460C871A4FD8C094DD26D144103590A9F4BC01EDBC94D6008DA48391A851C19E374"
02071     "B2249C5467BCB68E11A77BFDC4C9EED771C5CF16CF8CC3B688A7479D0F0B99E54F687E869C"
02072     "06018A87B75B98B87993D9C58829962D4588426B96FDC8892FFAED3168401D8FB01F41F591"
02073     "36C9286C45B352303141BBAE6520EC6620B645A010F520846959C4810190A7D4E6AB27A3A6"
02074     "78E21557A5127F15E4C6050806A5BB398FFFF75AD33F1E84249455A241F006EF15A02506BD"
02075     "171669CC296E66794FEB316DA723C7B255CF8CCF8ACAE60DD86B469A2C44C6725DDF13B70B"
02076     "940514FA95EE3982B72B511BBCD1FD3B5D19C681807DE3FF7730229675AA0A984985143ABF"
02077     "DF75FC7C350D8154738A6FC37954C5A98E1B672837BA729D3C645FE6E51EFFF9414CF0D4B8"
02078     "26D331020D2073D703C25C54BF0F068873EAAEA4BF39018026F1A8A3A73F9B1A73994EAD14"
02079     "9C8A3CE889208C3C3949C48B679E499D3DEB7930828285DE1B5E2D09F44C7121508D35CBD7"
02080     "598FDE0F7B4F8754D77E048C5DC53219CF8275B72410E79744E582911AD3895790AD3F721A"
02081     "608EE3B5F7A30DFD2BE9B10B1C5B5A0662E54C2531E54672C99AB3647CCFE1DF7DC50F9126"
02082     "FCB4E71627E75B31AFA829FA1926ABF650DEAFB8B3AF7A6692C10A3888C7795C3764949F9F"
02083     "12E0EA1577EEA3EB29AAB0ED64D852A36CD3D7C9151DCF64F00649F9F0299618B39E70CB22"
02084     "45E893E7808FB17B511FC2487572977A7068B5AA90E6518A603410A4C26F9BDCB529CC5F67"
02085     "41C660AAC3E37F7F4E40F32142331A1E8F3AAAE8A9E58C717BDFD237CB29AD5568F1D0872B"
02086     "111AE18F361D4494FB609F438F3429AE675B7C59991F745F7BC16BBC1581AB60928AF522AE"
02087     "91E8D5A801BC4FE8F9845E38627B01D14BB2FE8814D440D78C8EE4C8AB3771E48931DC3B7E"
02088     "18C7736DF123DA6E3BD0C303AC115FB93EAB2E2AA4B3A1496E35E575664E3A17E8C7C6C3D5"
02089     "4AB4E9FEE8754E85228241B6971F7863044AF204F67CC72505169F72E3FF1B8C9AAA8CE86E"
02090     "A01BAD7C87B788EFF66F721F5FD0A61E558AFE11652BF4BEE5664D81F8E2BA453B40D8AC90"
02091     "DD57E1A59F8F827156A65602DAF49BE12857E811F8066EA48A9014F501BEEE091F8E28ACFA"
02092     "7E20AB6613641FDEFEAAAF38B8BCC107EEC4244CB0",  // PRN 08
02093     "A7E921AACEFAB06F89B802EB01EB3A09871F66F0B85C95860BFBF78E7CB2F078BD27BE0948"
02094     "B42145F900CD2DEC1E08062960A14CEB9A8310A8C5AEE0E2D5F5B86A0351C83103C03C26F6"
02095     "7FE5C09253C11AE0DAEDAC78DA3581CA6F2B51EFBC187B0BF960B20AFA083D5A447E08F085"
02096     "A71F7B628838C0187EF71894AAF78ADFDE312EE590D210774839367783B7C8A538B408E92D"
02097     "D208FA7A9EA6D999DF408B5E41637A7EDD0DDBD69BEC7F9E39D1AAEF7B36AF6AC977EB0CA7"
02098     "83F50FF42482AECF9C88725AA45E70E3D7E492AAC74FF351668A7A2FEEC6EC2B3A08156504"
02099     "983A493B0A418A421172513DB2F3FD9057F42021E12170D3815EE52B5ED0F1A49896C532E4"
02100     "78A86D497585AB2153627FB1A9D87EC0DBC0621CC814777757ACE26BF6C7518CD3629D5A74"
02101     "08A85E1EAA4E3EAC99F2DC0A2456DF0EE69AA48A2B156539C0B2354F6857503325A0BE4C6B"
02102     "901B8D98505AF502577FC1695E039DC219224248D3F4C50C1A6045D1B5B02BC61F86ED3756"
02103     "E9BFB1F5B15919776309452E599C6B737EFDB8B0B1BB74B5F4EFD75DC98A4BF88BF5A4A6C2"
02104     "0DCD155E8D03E1519E24EC3613F30094EF287B27794F03C519C5E7B9113BED1F1718CEF5C9"
02105     "8A6A12E1192296DE7D806C77854E0A35CE36E4CFC9EC6411B267118D4FCA89C8FD9554EB90"
02106     "262D49D727E83928DFA27EB2FCBED2392901CF09C7E3B14EB8795A2E152063058BD213964A"
02107     "A5DD8EF99865255AC7CD5379017B96A6A379A736CB1A5D041B1F2CC43D27F7C234DF3FC5A7"
02108     "E4596E02111268781A30BD9B3A10A15716E7753BFE3EC8013D4ACAFDF6F2918D901CACAA5A"
02109     "A53FB31246E6D570CE1D7EDDF64065C4B5EAE52789E2D3AE39BC252095C9B703FE57507EA7"
02110     "769BC64341B5EA6BF320520E02D59493453837A219BC9851435CD6B8A9D902EEEA658EA28A"
02111     "C5DA822CDEFD1EA0DDBBAD07F0C9165742C8F2397884EA700F67F64F4D49D8E886E4791C3F"
02112     "8E29CA77FE81EFC90522ECA5F77128D1D64B89A9A8706D5C84846C9BC42094C402ABCFA7CC"
02113     "B147C8D2D99224CA0D6874CB30E3BFC23D77204F47C84E233B646822166902F5BF3070F839"
02114     "0188F3FA418C7EF1BE68B7550E4AC218E5DC9D7596B163958C1BC93F45F5E5F61CDFA9E883"
02115     "5A0DFA339D516F310A62337710BE7307B4CB2DE0088AA4E35713BD1DEF2456D1CABEE7D110"
02116     "BB0D753E85BC7637D3631D0E169055C822137BEFE8775BE0CB00DE8CE917AA1A851ADCB415"
02117     "1D3D6B8F7DDDDD0F57B9EF563D34D16E96CF3BA9AC589474218E5EE251294573FE1E3A0F7C"
02118     "06C380062B2F6F6C8B79B3042F1A68676348E81F1BFD90B43438462F7AAB0F9456E5F786FC"
02119     "7E812387CC32206085E7E21A300A09793B9B1B87C3F1178F5B6BF6360EE07F23D30293C5D4"
02120     "30826496DDBEEC8D52DA3437979904D7F25621D703CAD1E54BD8944ED1CB8027670642B425"
02121     "95AEF82EAEED03F26A4D1D492F5419324B8B4862C13BFD4F4A29A35D5218BB2D5BFFDF85F4"
02122     "F9D3D0B584F096922B3500E7A1D26AFB4616D6C144C540280DEEAFB721F01B109EE4F5856A"
02123     "E534B1ED4734F880C9EA836EF172A8F2566FAA576A80F2F4D670AC7F8C5945C016A16E3C92"
02124     "795AE445C8536C6C12A3DD65C64B0844AA0488C109610FDC88FB21791663ABF41C63B3B586"
02125     "B1A120E55E2DD35191370BAF43BDA644AE9D7E47FB0623CB653F579B96EB22732F833BB0F5"
02126     "9B217F700B8195E9CA4CD593939D5824EE98E5D13B865B1375FE7986FECF8B1592077D6E6A"
02127     "446F420CC9EF4CD934F75FE3769F1EE29D0545F748",  // PRN 09
02128     "067E55DA111785828DC1EFCE60B788608ACF2928466F425975E29FC11E3EDB16B37D93C135"
02129     "5B0AA7D3B0D906C0213105F1BF8A2AA3ECCB1AC1720CB783FAE25B154FC507B725F2594E1C"
02130     "671E20D549718AA9308EB43E8A71FE15D7619CD2EF731CB309DF97E0404BB10AC9D6458382"
02131     "9501F02E879DEF1095E726F855AFB15EA1723E1DAF883A7D2E0AA3999E80B0489E7589DBDB"
02132     "9667A99A1FEFCC7BA74DA490F9D63623C51936A901A74DA1AA2C3242055C62AFD9671B75D1"
02133     "CC3D7D9F38D9F867E7AD87E888A0D85EA1B452B7DEAB98D9D34245F1A60AD639EA6E9306C5"
02134     "85A00D8D026B0BC2D302E27D621E9D0273B4E782C2A46EB7C069C0FFF44D98047482B327D7"
02135     "1D43278C0EFAA584FFCA492108FD3E84D83E679DE5EA5C6AFC7CBD3F503F027B1C689F8228"
02136     "831396F7ABFB4B2344DF7F1835A1E7C3A86C459F0F3ECD643FEDCCD5CBAA4CFC9D9652CCF3"
02137     "1BBF4D1E2317B05397E17850B51461B53AC475FE2BDB137EDB825EF548C78B80F5B08A513F"
02138     "87462C94579968C36262A97058C613C4232BB9D4D9D2CC939437845245FBDEE1518EF2F6FA"
02139     "53AF66F48346C4F69133F70A24F00E9439F7869ECC80FE008DB7D19C4631ADF628BD659FE4"
02140     "C6753B96DA4DAE510215A37203A1EFC26D910317F18C16BCFD993514D3A3AA843463DE914A"
02141     "D15FF3D69ADB9A903015342ABECD807417B3A980201515A8E4A05CDD19B55C055ACFF5F341"
```

```
02142     "6617CC012CD9FEB5FCA3210291780B8AF0ED41EE70DAF229CBA8B7419931AE4CC02FA1F8FC"
02143     "C912B330BA6F0B771223EEDEC12E7F9C7A00E774D107B7F85FAA4C76F8CC578C885578D99C"
02144     "069F96D1A238ABEA068F9CCE949177A6E7585AF5A414BE9FF7AE6E2CBA84AF6B329501D556"
02145     "317EF7B3432003B4DF8401A4F4D3CAA38BA9DC73B66EC98693A6F8699AA3E1CB5DE6C17232"
02146     "DA689F469F1CE489CE930B5AD949127454FA133354C030D5C0E61266F49F3EC7FA68F88C71"
02147     "A6510E8FA2F2177A7BE2DF209534D42D135C5B420DB82EA37183B64A6C30DECF6C5DD6F193"
02148     "0D2E13DD6BF36F3627729EA06E898A4456F011B0C16A1E800FAF35516D7AA1B513FCADA1FD"
02149     "82372615AC3DF1339BA4D7BCC9AC12E5091672B179A3537A05418F98C6C614F3C88D77FAB4"
02150     "779F7CCB36E5DBACC93C9F1133A7C5D1079B7155A39CA81283DA09A6F6D0B101B4F2F0BE4B"
02151     "1DA059E9739CF2AD5644AB8A18FBECE9037029DEF778F689EF8E0CA6DD61476A9B6657F79E"
02152     "8FAB1F7C02ED687D1A7F1E4B2312A13A9D0B33F08852249849D9F61AD40177C049099B4153"
02153     "BC1CA4E454E05B7E8DC0C8DE5BC80FE64132F4BBCC35A0EE4B4D0025C5C3507A82051DA017"
02154     "4023C956E040AC37CA5473D73B0F909EB11E4010FA349CF5023C590C9014F39C53925E4D71"
02155     "74E5715A3935ED37A36250662CE41BF9478E92518082C34688D485131746BC848A96D5C797"
02156     "519F73CF9D13821181EF17110C1F12DF1D4F9EA234ED3A7887322A9C7E1616C7DFAC8D3089"
02157     "302298D482B9E81FADE4F5CEE785EC9F4510903BFEFAEDC54BA621BC500E6D3439684243D6"
02158     "4D7E00F1A9947FFB41236AECE4DAA7B03273A47EE2C2FD872685FEDA7DC87F0C5F16342888"
02159     "C18C81CA695F72B5E3979DDE44D546BA0D3DC928B56479363AFE814C59DFBE0EAE1A323D02"
02160     "ED77A221EBC4F43E4EF360DFB50E0E5B60A1A9CDFA3F72AE81F1A55995FDE8D15E38BEECA0"
02161     "B9384A0665515615DE46E11D3D96D199E32B83DB7CAD7F6A6098634D8D207628450EDC8ABB"
02162     "BEC233E2339320D1AB80F021AB10059C9B6565E73C",  // PRN 10
02163     "3E4B585767C32759C804C1CE4E2B0E4EFE8624004ACC2BAB24CD0C40A72E6E9CBDC0F51DAD"
02164     "8D976B092B0659C1F8E96F2CF6C2F1D7BD0C46639759AC480AAD7BA80785194FCC778EFDCF"
02165     "9B760B5C0C3F4258FB4F6FF76BB5C7D81A37A04754110BF2CEFF14B8DEB0B982F03A906429"
02166     "08EBF01F1374B39CEC341347ECEDB9EF4A5BD2B4699E6C77ED4341BCAAA32218B92384DA89"
02167     "A8FC918E0AD32DA754F4C780F9AA179E0EF9F5D1776B250E41AF20ABB11F3820A976E3C92B"
02168     "3BB784E5122BDD2FE5821C2F1DD5AFC0F7808120D97ECD467A7456F05ECA0D73AB772A5441"
02169     "6C80A0491ABE7703E7C63B39A071262DD1A1B7CEFB57B33F190C1A21D86A6B73021BA4A805"
02170     "B4D78CE1E1AD6613843EA175410855F3EC96133C27E10C8C3BCF12E285DF6BFD2AD6A22146"
02171     "B35965DA2675DCFA085B3CC6D817AA1A511E29414FB4DB132D9F123FBFFC49BBE3CD0BE0BF"
02172     "8736E75DD36680303A51071467F782511CF994A178BFF0F192416370FC5D8AC58568A57F0E"
02173     "3D33F6365A035054D783C833E2C06CEC5C2D5A541426486A38C28BCEB8360CBF0269B1A433"
02174     "0FDB5C5C2138FB44741ACD7C1F5FED51077BF0C9CE863B5AEA08E338BF651B8223812EBD57"
02175     "0D4327131534090320C02C3AD92B5BF0DF4BF5E006156D05AAEC4045EA32EFA8A3CF32D0931"
02176     "70C887A64536AF7D346CD90FDF91321D1A63E658DE26C2779AB934927B39776B5495D83B3C"
02177     "893C2E2B9CCDD599C39A22DA4E536DC286A54B87C778A548E4BF543ED5A561CAE61DC49016"
02178     "D1E95377A0DF9B3EF840F698916A0043C24A2A49826CD040AF15699C428FDBDC05FD35AA9B"
02179     "34811D9DAD9D84E2ED9FA2A26BC94C27981B4A0D9B4E9495919DFBC2A7B3D786945480E7A0"
02180     "7511A453C2691656A7892E6A4C6686FE93BD310C2C25FBB9005B60C4E4C92C0E4DF6310B44"
02181     "95A0ED2D8347B221B5B6FEE8F5B7BAC922AAD32E4D245B5D752E2DB8BC5304D52A0E7EEFB0"
02182     "BBCB4A39AAD896FAB9926C6045D9B4BF371C9CE12E3D30C730B4939EC6ADB76F8049A0E4A0"
02183     "68C55918108C61938BDAA830CFACCA00550E1431EC94359DA47F6A05CB82F242DCF6AF79A1"
02184     "098CEEFCAD8884BC468974AED85B2A2847E093A45D88FB27FA1E7602653B083C70BB9B7A2C"
02185     "FC3BBC4D45A89EFD09A22628D8B039A6247D46E35BB37E604238128220BA711475EC497D822"
02186     "7359C488955C8319572F47D419A1945E5EA628BA9F114EAF8F565FA95110D273411D01A7A6"
02187     "D1C96CD60CA84DDA156805771411AF3A4BD4CE493D9DD95DDDABC03F830B372976AC302B7E"
02188     "F0038D93978F63F1F25507DBDD27EA11E2951363F455D24304B324BC59AA73364BF397DACD"
02189     "B75173575D730F8F25E3394F797CC2D38FAC26991DC238135EE55FFF9C81CC9C828FB8287A"
02190     "B72F33A28D8936D8980C221DBCE786D5141A74893B426C6B58631E96B350E50A7E664BFACC"
02191     "7CD4AEFD366EE11E89FC4454F721CC1471A80CED1BD44581E5D2AC177028D0C6C7E559434F"
02192     "9382BD1766679685657617DD8554FEFD17A22FE9D30C80F485B46AB07F43755CF5AA13E827"
02193     "0A9B3101AB0196246D87394612DCF980FCE24FAF4D10AC50F67FD00B4EB29C29E8957BF830"
02194     "DE3E9CA028BE889CF0BF3B836D5542991B0F28229920A393F57F6565E0095821D296B3AD4C"
02195     "C50F66D9B7B70C8D3033535AD74BF2A7A5B67B265FF7315174F67F883DEDA2DA30CEA7BAFF"
02196     "05519109D7301DE9F45C0B7663FCE41F88ACB224FA0F2FC954533E3EF633D568E9C201D37F"
02197     "3D7DE60DDE22AF138E4C90C86CF6D56177AF8A23D0",  // PRN 11
02198     "7D82FBA23D20FAAD30DCD0F0A1748BF5E0B0D80C8A8C0B4313AD2B92C8E9608363707A5C28"
02199     "9C771D8772CA9C5248C496C5F0B59F84DE2DD2F6EA681E7BFC6AB426679B19E53237FCA8C3"
02200     "1EC5534695F19640521213084FAFFB6CB9E4200EC253AE7F7E781311AA92DB5420431E689A"
02201     "ED416AB4A84419E6CE801A6A2CD67AA16A653A29F7F9CC4D26C8ED3A00B5AD49B06AAF282F"
02202     "40233E8A84F944E8B5D8316E68B9854A4B69A4F160DD2562EC4A20CF5AD65D8656E70FDA9F"
02203     "35585F026F69CA9661D908B605DE4AD01F3383C157304FC0FEFE5F8F23CA2BA9013C9C0914"
02204     "CD3948BFEB270BE3CBB37575337CBBD75DC8E8A38F6AD986BB8DA4F3763CDBC2534050B812"
02205     "86B5261025E88A6319E2F7B655D37C9B2796594B68AADE3AA70B0DF5E2C2E135C4150C8877"
02206     "624341A4F1081A8F77834697C994F546B1CA1F7494E6507E3A8BAA3F058E510232E93F408D"
02207     "0A912E196D2886D478872F2E926CF032468EA7CB17C28CF6F520C805D97EE714C275D7F49B"
02208     "695CD9F76681A575D7F82B95F59EA3A3AD8DF471011D6BAD17C5E4E93F26EB88373143F2DE"
02209     "87865B40A335993F9838ED9BAC0873FF70D3F00F9D23496B0D4435C65A6D39458F269C1A4F"
02210     "541B0C1E32BC9DD3A24B8330CA81BA450ED3BE5035A557296021881EA2DE7C7D137020E8EA"
02211     "416A3808497D09AC3906E4CF9873C1F6C44803B075EE575792075B8C2EECA9120C46966723"
02212     "DC6A508F4481D27AC0F70BDE28B627D12227FC97A42D564C37AC312DDC1B8D746E4885A0F1"
02213     "1E9F701237B4E38E678D2B42F5343F09641887FE3F33DE1D40C6C8CD655A7600D75CE72D93"
02214     "19C286836322F3B56C39BA78933FC19F593086BBAD74CBF0A38CF4B9116AE0E1719D339EA2"
02215     "5D48781A0912BF87FF9C3393210237CCE19B9803EDE13A37B69A9DD7AF46A39ED501B47609"
02216     "062ED00A8FB1A4F3DDB3EB2B950B020FCB7DB9A67B4CB903A2ABA8B6544AF874D615F0EC87"
02217     "69D8AC8A7EE6E50D8EDB152FD8D4FF9A46CC7643A7C5CE0CA4107A462812387B7A7879B8A8"
02218     "73D03086EC46F276D903781C3E8040CF79D66626D172A8D51C22E97B8FD3B17CB84FDB27E2"
02219     "629777B1D94DAD7E73C94C8C9B954BC6C1BD4267223C2C93EBF9C297E049C19847FAA419D4"
02220     "3364C1B52009A377390F9E11B007185198D2CE499BE3D2D0BEDC1DE1770A886687E4BE7EFC"
02221     "FA044034CAD17E2B04115F4CD03437F63D49A9A6F5D30F8826DA3EB010012D0DA0363552EB"
02222     "7C578F14F03FE7331B594741B524632683AEE3790309657E71EB398C7E6BD92C8BA073CB83"
02223     "98ABA40B59D8DB852FB6B140DB2321C8F3285CE1F17185CAE4879B86BB7F42422BA02224A4"
02224     "9B998D18AB11BA768ED42D0F3ABD270AB08DEF356CAF62B2B27289372F7B6E695BCCB85C1D"
02225     "26811175AD764FFBA8E4695C26FE09D1627366511BE86A6851A1D395F0E56742EF65557642"
02226     "68AD3FF9EAF9B3B4FC5AC7E4A6F998D9E60527F8F6287F1757727655F4BDD1D6C81D4F47F8"
02227     "6B2ECBB8271AC41E8439C8E1AD5F9F8F394D0C4DD5D956CBF102E6F8389F9669CE7197BCB7"
02228     "A43A46FCD4E7FCAB99F407A6C96EFBE2D2714DB68025B917F40534ECE2ED6DF5670DA523D0"
```

```
02229     "616757A7A0D21B692E3E202BB4C8DB3CA0E95DA502216543F0D0D41DE560BAFCA5CEB63CED"
02230     "63EFFF5AD6FB32BCF66734333A12B246DC3FB58E66E3A7CD48AA1378E1E19AD96BB1515C28"
02231     "3AA687581FE888B5DD5FAF88F5514CAAB8D017FBF7B3CDF4AA040228ED8C42B6675D9EB39E"
02232     "BCB7533226B0C9622AC1A3EFB6DF3E0AAA65622524",  // PRN 12
02233     "E33BC2492ADC8D8ED6EB66088EA2F93402ABFCF1FD2C4DA925CA4DFBA610FDE285B00BE829"
02234     "F79B438628B22B0F9EC5F2AD990DB9E4067EDD7A9BFD07A1873CE41E39F96A4F3C72D90FCC"
02235     "09DE6F7196FF96968DEFAABD80523EF8CBD20559C8134740DBD3793CE68DF223E32C26E7E5"
02236     "788E6F3247A1EE456967C2524CA4D7EE9441A3B59EDA8084D04297E0F7C717480D590C90C0"
02237     "F769A6C8F7AB09D607BEB8899E1D6316926F57FDF5E225B3F1AC45C4991C1F7EE25BD435A4"
02238     "5B2A24926CF4E6C5F53FD00DC5716700A8A806654169C1340E6062D40E81F69BAA41FF061C"
02239     "DE6A0D44D5F9C08F2C213A5A0A0342B5BD4E63ADB1ACD89EF259D735D59CFE01B79E2E22DA"
02240     "14573572F4FAE831AB5D259BA3BE4D5535DD556425B2B6F6655AA604A527D0C5C680E5575B"
02241     "C610EB52F7562A41E66897381BC522964BE4CE47EE0575C9B9C47A871A3C4C687308C5699E"
02242     "22377330ED2CA5C24A662502D6B655FDC76A8E72C124A38A79A1584BF10623EC9E062F47E5"
02243     "A96F5C95237D59B0C02AC0303DDD5C56AA5725D9115760C3E1DCB2B6834F30878550080EF4"
02244     "9D6C301EC9001F541F09121D0E58D0CBBB8E83748E8009C7C1996C02D9D2ECAA629713D86A"
02245     "987BE5D9D46C603F68A447DFD8B155CC549519D3065452FD32FF9A298F4CFA85B8C4947529"
02246     "1FC65E2B50CBDF1F695A51648E7F30220DFCB8A981098729DE1034D7884185C5FA6612FDA6"
02247     "FBDCD4812F38F35BA3A53AC1271B4EF5ACF5D6A3D2C026524B2772A64E8DF987117830D2BC"
02248     "D86E6D0C11C4B785A3857266FF84D36133B07F7B17016DAA576786E2DF8552E2A893D33F95"
02249     "A0B95CF7B145729716251EAF5B0F9BE02C768C04C52B25244366D2522FC80A305DEFBFC9ED"
02250     "EED9B5110C8B580D7AC7A4AAE4894D95B634A29D196D85B726571572EB21B652318AB5CCF8"
02251     "BE4A81B1F2AE8A4605006DA8DD19A1CCC76CE460F6701AEE323CDB04907B7395E5EB710F6C"
02252     "CBD2F4A935ED085B4A0DD5DA0E13C8575D45B76FA9682488F742CF3BE1E3701A7C3107352E"
02253     "A22519C0BBC09675DF4582A6012DAD8931586D572F048C7290952FC72799006056EF5C5C6A"
02254     "AB7EF5A47142EF1A6FC76504D99A383634EFE796B306E0DA5CA3D68D589FA417E6F6BAC025"
02255     "07247A37BE475DD0004E77ADB571F2E39D72FB0627F62862C75D95652BDC0AA2320D991123"
02256     "3E6F30C2797961E538C6D54DE9655212F3BB6BA5E319ADF99D5F62E53218187BCEE6F6AEFF"
02257     "766301BAFBB65A3EA83F4D069E001A33E5B4280AECF458A9784ADD5D84C6A8C271DC1FC31C"
02258     "EF0466D6D6C3A5834F96A134C0933221C4CBDBEE1E29C26A260F6BBDFC53A420E27834D850"
02259     "18A23A311A81703C304CF416C1DEC0CD8082EF60138F5B983430509B223789035798116E262"
02260     "B48C0F2FFC9F8FDF63C53B9AFEBD8A05D862573A73D4AD6E085306D8A538FF17D2740F72D4"
02261     "88B0DE5EFAE6C7FA336917F9C33C4E9AB30E6050728CEAEE0406198276FAC4E26FEDD93C8F"
02262     "23A702D205D2EC8E8FE2C8D7FBAD1FBD4CA9BE9EAA11A0C22B059115CB21376212B3BA36FB"
02263     "283FB474A0BE7DDB5ED346E58B2BFF5D6CF15EDBD50062395ED5E8189995A14B171B9F84F6"
02264     "28E1995AE6E6B4DF9420957765880455F7C205FDAA8C9F4BFB49E5489D82980A9D8058178C"
02265     "E9826BEEB37B178751C2DD687470BDCD6FB170EA11827099D0429E6ADE8A243A510EBD6100"
02266     "B26728F66F8293AF30F6EA3E7EDB10AAE5E91BA71F21D7A4B342D1CC4FD365B5157CF5A783"
02267     "C67AEA038B8B50D71D7F1DEC056FEBB038F4D4CD84",  // PRN 13
02268     "31372C958C1ACEA7FC11056CE94F76AB1E6BC0F0A8D0CD4D8FD3469D1625CE7F2D815952D0"
02269     "09D377FF41F58C0F0DE977E6826CB7EB549699BF365DF4BD9E5B2F4B0CAE1C2E68869C67DE"
02270     "525E18F70ABFEB47B6D365FDF5C529A1C68628476A01D5173E9EA883B884EE7731B3E4BE08"
02271     "A5AC4817EF2ACEE72948AE5295FF1DEE5D96317ED325A35CF72A8A3F4009EDCC222A0426E9"
02272     "C559B102D67A6FD3C4A2258C931477046C9557B040B6B2DF4B0706B748F1FC4C1B9B8CDF38"
02273     "C20C3FA999EC347B2FD7AAD8B634647D0DF4AB0423CB76DC2E4E06115B627E1072A0C4BEB1"
02274     "5843BB16046FC01BC2BFADFF673ACA8B938058B820715FC3A03A75019D6150FA3B74D1922E"
02275     "B67433CC4D3B405DC0EB808D7990642534A9B1555610FE9F253FBD4B37AEAC59CB4860FB0B"
02276     "B3FBDDB631E2C42732B66DC0150BE1C0A1AE889C4C8B4F2CAEEED25945BE356D994AC59E59"
02277     "6E19CEAE72CFE49E004C534558E85B238864AFBCAF4E126ED47A48F1FF3F27F65AF1F703A4"
02278     "6609F3A87F9209B14C8D8AD45E13C5FB579D359C9E0E2FA6AFEC0006D5557F96ACF1149C75"
02279     "A7DC581458F4C694BFA0F513E9C51BD33B8D627960DF0FDC022F0F47A68EF43D2F35090C11"
02280     "18F4203E03D037F4B9F3D6F598A02912683EB26FF4E5F4618AC1308572054A1DB70292AF7B"
02281     "170B94BF7782DA71499CFA8287D916E82BEFC570FC192DABBFDFAA3BCE8AAB130B072D661D"
02282     "4F1468FFBA745600CCDF84DD9CC4E9B70EBA7DF86B6565F74537282178B064F9FFD7339E8D"
02283     "6D9F80176046E01CCFE906D835E0F17FC4DDCD15D8CAFECCA563D6516716C415FB316315AE"
02284     "52639595D5A13FC509A6972C4662181A0AE1931BE41ACA2EB9FDF54E558613DC704B6FC3C6"
02285     "6818CC608158F6D86AAB756E5926469FABE491645A95B2AB2038B15526461E2ED664DB5E0A"
02286     "21709B27B84D2B9340139C554A30E81F76B19B7EDF84795C8A2CED2E43468A00C221D4BCCC"
02287     "9230ADBCABB4B8BF647618E09F637351A1CD15E9D5A156604BC4190CD37999902FB56DB55E"
02288     "E6A2850366CC2DD659BAEB09690E62E01A5288C4676A0E0C5764B179DC69CB222DFB3D071A"
02289     "D9AF9BE79E3BB00B3A47C45DB863F028F216BCA4F0B4A089111EC43651E71B0C1C921CBC14"
02290     "36B75170EE8082D2389BF32FFA22EA21AFF0D8F8DA347E0B2DC6CE58EC59F38D67DF8573A7"
02291     "532D5B5A40E995BF58BBFC35AE3DD3E19441AA79E8E3B57E5CA5B3B5612984D4BE0F4D6992"
02292     "406F082D499520A1AC3505A98DC0734D907182680434FFC2849884CC785944071A2F49AD34"
02293     "6F83E51D64BDE61017747B81834A1D8C55AC2EABF59B446ACEC048E1B9FDE5DE103E5508C6"
02294     "9B96092F126DA219800DA01AC7CD45578605846482AD5DA52D2C7868003E302DF2CCFDAF65"
02295     "D42FBF8667B0D589481EC3EFD3D8CBB15D6B7D3853CB5D42013CF78048667DED7743CDED6A"
02296     "F0E0C89C1EBB19BD7B64BC55BDD523822381D99618FB5DE6AD2744C099003B2EBAF231DAF2"
02297     "E21066D6C18FCA8ABC78A2FD75A9EA7483F1017D130E917F9CC4EFA65D2E27BB35580734CC"
02298     "798C8F47EBC9B13051E545682B20A03F9CFB8D1E6FEED2327D96A5D9AA9363D36B78132C38"
02299     "FB3B2BDB2415A742C452BC5A5DF0BC8C167A1D8DAC3A60A636E79AA6B12109BBB7380A10C5"
02300     "14AF30BFD91BEECCEFD9A4AC0F8C13B543860FC5AC9F4FA711819DFFB4F5D51E9DEDF47B66"
02301     "FDD37B20B839067F3D1F7723B9A6AAFE27FA813576D44490FC7D64EB7D35AD828D185C5F85"
02302     "4E95D92BD59502C0D8DAFB337CB49823E474C824E4",  // PRN 14
02303     "46676F28487E3A66F6C502CDDB4171BFDA4AC74950AA641F771BDBFB32FE637520DD326017"
02304     "D9557893B96FAC52CCA65099D1F0171F8A9422CA6F5F24A7398525B65592DCEA2AA8F726BC"
02305     "08AA8B40E55DDEFFCD8AB1F212DB4264C61A80F59866AC763F15CDD280592CD91B22E824B3"
02306     "92C429E9E52FD4A07E4D6A8C1326AEEB7E03A3425996AE394EDD74A1AB8D3D3E4AD7529E1E"
02307     "B867870699C1B832AA65A8339024670C56988FB9B42D431CCCF0AA949890B83406381AAC09"
02308     "A6E034D7E3ADA9A0BB0F62BD936B281181091AF7C053715C04F1960DCE215F7CA4A662CD04"
02309     "5E69EC493BA6C694EF97591F755F5A22F0EC7979CA2C575B158E6B621E4DA9909DDE3F5E04"
02310     "28FC9ECE064B4EAD13638F375EF7FDCD3A1B53BE77BA7B3CF17DEF01CB07166E2089437191"
02311     "E003968B8A4680C387A709D45D0E0C40BA85F1DC55B1BD75C7513622415336D66FB31ACEDC"
02312     "8078D86B5489E66744767F85E08BC88329C58D3EF091526762D0DA1C686A4F9820CC4B65FB"
02313     "5E02EEFBECB6CAD6322E9083854ECAA1F9F595F32AA5BA0A83B856FBD150B30E064D658D0E"
02314     "BBFB2F01E831F05EE59235017416C927962A537EA506C8BEB014BFAC0CC2807BA62ABCD360"
02315     "85ABD620979457236ECBE3F815D6FDB0B37B7741F4CB68E7A4B579885A09E9744553BD3CFA"
```

```
02316     "AE1B2135793FBCADD14A674E5D42C9B72A361D1A21506570CB8E6D676C6FFE08C0F76246A0"
02317     "0754760753F181B31F23ECF61581A77FFF71BC23A284A1CE88FA7E1DED0B06EEBE10131DD5"
02318     "F364889F59AA35FB6474ECD80474182348A41E206773BCC41488FFBDCED0D243F1F22AF78B"
02319     "664007C5F2F732C45B980C14527724FBD6122A78F41AB60F0436158D2DF0DFB4DFA0ECD12F"
02320     "DC5B96EFF1490E64905CFF97230CB4BAE3E6BEFF9D00C7DCBE6E8586B141CB7B49CDED4D0B"
02321     "D9B0406DF95492C1C4FABCF88E283D63B7855FA79B4616E7A58EF87E0F7F79DD4AF8B16612"
02322     "BE175ECBDD2DAF30B6DF8C4BF28C249236B6E10181E8A30B3CF0B17872BB92C0C972DE1A83"
02323     "3342EC8550721075378AA1FA4483EC77C31E0C87B987B8418E9DC315022941345AE94FD9F0"
02324     "ADF99EA0E060E92776DB4DF7FB93811891BB0CDBB466436AF538F9D70EB47FEF93DBDF8191"
02325     "822D50359E58ADFC092186F5582F7019384539999 98E787405B9C8BB6C9407099E73707C3B"
02326     "AEE08904130ED6ED91E7880F069FADDE26A483439ED81A9D999B3FC3362B8211E468F25B36"
02327     "B93BBE5942A96B831FFE33B5080F0AEAC25F9508DBA5EB924B01AA916FC99EC1FC7C149C7E"
02328     "1F0EB5136C2D15886CD5ECF5F395F0B9B8A85746D0FA18D8A7CD98AE6103D609CF404C6089"
02329     "F91A22411C378FD1FDE28BF8ED7D517CAA3ABD675A7276C3655A30626989927A93E3EA4C1A"
02330     "98BB70004182B0E1A206380F94C27B2114811E20159B193DDE84BD4D7535168ACFB372C880"
02331     "4AA344CC931354CE7C56A2DEF1DA86AD2B6AC9A8740405DD2C5807D0613F6104B094024F1C"
02332     "FF271E3B674E4BB84A3CCDAE9528A36161890C52DDB62433C1DCFB4B22B46C890ACBFAAC61"
02333     "07E64282FBD9414927AA8D1A403CFB693480A8EBDDC22C9AC0E0F519B78A87B8E3CDDB13E6"
02334     "B3F711C9F4732121054F261EEBF8960D96B86D3EECEA8DC6A4C3DA6112028CA5D3793D3511"
02335     "BE32591F350F98D9DCBC4FE6CAF76CBBE62AA7F33C3E6AE7558263FA35D82B358046242432"
02336     "5B2B28D7773304A76194CCEB50A71F8BA827AC9664C5B3A83133920A6811862AD0E703A61F"
02337     "ED68C5E46DA3EE4040515FF3FAC7D566B5B4569DDC",  // PRN 15
02338     "D2613E9CB156ADAF36198A971E462F0CB406AC07B6C729F0D89B37170790C7D14D671A83B9"
02339     "8827D21E5C978BC292027096AD84D5F4537DDCFE7A9B86005DFDC056B8762C08364E118DA2"
02340     "42773A4B45BE4EBE61C5CC7656C7B40E35CF4E00987E66029153F13B1A86F0781F7A1DB045"
02341     "C62722C2E04E70516CD77B453B3505EDFEAABC0FC59CA4A73E330D7D1A053D005A09B76E7F"
02342     "045CE132CE3479BBDEBF899ED79C6FF008E5F47CC8905665DE54AB835573728EBDF731B660"
02343     "6D63DE7D2C7932A93BF302D0364BA4CD284B6E6796B17D36BA910C6FA2833E4B7931B3C7AC"
02344     "787515A7FFCA747980814AE6CEE416A937052DB39F9AA0F0F445D6986A678D6A2865A50A3E"
02345     "781D9B667528977AFD897F039DECA2D3EBAC4B9B0E2CBDFBBA914943EAE860B50665BFC8A7"
02346     "8F23D380B1CFB9ACF29D22D6FEB121EC2CA3BF988090454D5F60E0FA7EF15FA79E4109D02C"
02347     "2FD16A1A932D07F986AC32854233A494BEDD50041DC054AA7E9B00ECD8CB929B1F5874E68A"
02348     "9E6414B99060B051FC50766AAA67390050913BC937F1EB359C61246F4C6BAED9F7AFFACB35"
02349     "907A2C6345022FC7B7EE9F5A4E1357020773B6E5AB442F0263BDA8A7BCAD48F18E4D70444E"
02350     "0E63BB43E4BDF9BBCBF3A55012D2B5C97F46A78A4EF307113E43FBB4AC9B59DA1C673BD4DD"
02351     "426438DE9680B019F6FB215BEA45E4A1CE08EA8FD0453D7ADE1872FBED83919D73D35962A5"
02352     "338FA72CE3660A42A085A691D692BE4D4B04B05E52A72C96C0DE8B2C61282ECC81FFCF38B3"
02353     "6DCD3D2C85D26A699129CD1E2C32CC05F127872F22D5BB16830C8A35D1E6ADEFEC32955603"
02354     "1A5FAAD0F7C42B0EAC9697A824A4443B4F69084F5DEAF6AB8EC7298BBA1783EE798E60DB59"
02355     "6396B376EE80F0A8C6E3CAF1C184BCC2DF5D991BA43549F1153AB8144B98DD041D9356656B"
02356     "D73CB2583D0E76DE2BDEDAE587400E208665D16AB91D12F2E54E994656DF11ABFAB1A34671"
02357     "6F869C44EF0D34A25C8AC2FF22FA2EA93C2F53F4248A1EB4BBA2672C69F3C2526950DFE581"
02358     "F3D41390DEA7F85A9DD9DDD3BF646976AF30495A5DF3389B345A2D83D3A049CFC68FE8A280"
02359     "276B8B74CE9F46DBBB8AB3799C4F9A6ED1F74909D4257ACC5790199852288A646DE04E728A"
02360     "595147DEA7385CF19B5F3D257B5A6B2A8B718EE5573E8B143D63C80E9ABB07A4A228DE973B"
02361     "1CD4BDD8003EBAC41621B0C58A7290750EB8AC72A70D1DBF0B48B848BBB59990636899 34C8"
02362     "65E159A7A06352980D55B6957C90D83F774933074F1E0A25D746155B884B5DB60E87C54F8D"
02363     "F312852A3A8AEA9A201BD86455F10BFC85C52CFF8DFAC6417611EF75C44FF7306666D339BA"
02364     "BDE9D9136F19E3528464C1676D15DF8F98BF5C827B0E2C9D08ADADD550B45A7B5D75C38669"
02365     "F4E69057D76AEB2C60D4332353B8036E0A2C8885FB024C836F033944DC97EB15A0D6C4BB49"
02366     "6D34D8736FA2248D39D1BDBDF156A2471289987866E5F7B2131563991036D06C39A52B1643"
02367     "1631E50DFFA9429DCC2B6B22A14EC249B471AFA0454E257935336D190B61178A25D39DE494"
02368     "EF31ABAA10313C5E4F33306BD2E73ACEDCDCA732D3770C6A3DF47C8DE4A41378E1DC500A96"
02369     "36E133A4A694E1D31193CD6DFC46D04D5E9F79028E96A3FF589A65010BC03D872D075FBEF8"
02370     "CBE0BD3ECAF51B34B46847C3CF4017B74D0579C7D1B9B3D7F07C76F740ED84009A8511E7AA"
02371     "A4E879FF26A96109D829453AF0F179E504C5ED498E324570F6D280EC586565D9B899F5C368"
02372     "0F1F4CFC1A6743631390C60B782DFC09BCAFC0549C",  // PRN 16
02373     "EB443CE9D186B88CDFAA0BD64D8832C38F909D07B105322C669700906B564D2FF5563418FD"
02374     "BD8F251F431C446DEAADE73402A68174718460872DCCFFE53EB0F926C52375F2A52F21922E"
02375     "4FC39E5ADE2B66F0F3CF1037DB90781D7982DA4CBE85D4F8B006F5F885BB28F58927B05287"
02376     "B91055AB71368D7BF59DCD9BCE493C9A41E4394B5ACA13D474B447613AB0336C21334AB321"
02377     "02C482592DEE4F58DA10BB460F751264C0A1C99213F48BFA497266675DD215FF9ADADF1125"
02378     "0D37A99899EBF660F369A6EF3158A2F7F34C9A2AB88F4374D94F199A94CD06BFE5026B9CE5"
02379     "D86142AA7D38976227DCE0C617479FD2691339883FFD5277042BEE94B52FC2F0E7355CC5E2"
02380     "A5387E58011DBBEF3ACAD589BA0F59FDB93FECF8B4D281A852562C33CA8435137D96958732"
02381     "C4449333757D929041FCEA8A092B54CECDD5B26191D549B4CD853183735D31C2AB921D7724"
02382     "FAD18A939C1480439FDB7588BCE0FA44F9438252080B723AA39FFD633BF6EF6CFECEF010A2"
02383     "5CD2F45C32BC378F0EEE54C1AB258BC3A3010644C5478FB2F8E182F4364ED8CB06D1CD3FFC"
02384     "DC824E006354A16638509B2BE51A539CA1423280AF03D2F9E7117B728A1B337E85BF2EEFEA"
02385     "36A6B399606AC9C8EA1B1190E8804547CE454145BB70BF06CA72F4375939F97C7F5E65350E"
02386     "35EAE2B28E939F7DB00E1F53FED4BA864EDA16645B11F8AB5F56A5024930B2644C0AD3B543"
02387     "9F53F62DEC619D265C0280201327D4CAF95BEB7C9D4DA741A5D28AFDFF6E9647DD7C32DC3F"
02388     "AE6C102AEDB45678488D867EAE9AD409D62CB3222C03B234C700F007E9B74BABFF40C81CAA"
02389     "CCC5A411B0A57838F2223BFF9354AE6185EA61454FBEF2EBB127719965F53234C48B959136"
02390     "E6A9CF864EAD75607D5F4461DAD3E43F0D87A83ABB46C525FA54A946092258DBE86CCFCF54"
02391     "06F049DB53C67F569192D282B3945B3C8F8E42F40862D580DB2DE229D4227687E4901109A6"
02392     "1582F2E7897B5234B31314E14EFD35E92A919614312D95C166C2716AEF68403464D5EF1D46"
02393     "EA858E376BC2FBDF7BFEE7735B3633065E41FA992DC3D89659A5F6188F5F959681AC8AF7F0"
02394     "8365188E9429E40F7A782405AABC129928ABB951BADB25B66DEE641BD42FA0B0920207EB4C"
02395     "CB658B41CF9FF15EEC749D3EA86050A470052761700BAFC7D645A090E5D149C16FB8B1A7F2"
02396     "4F24B0C9DE58EA782A859F4EC1BF2E86B85F8BCBA734CF7A3206D5C307BF6B4D6811216E89"
02397     "0079B6E8A980872AD9E865BFCB46E8CFF67C0758FCF3E693193F31A97BDCA220208FADFF6E"
02398     "1F7294AEA7CDD6F01A19845A1816E2500CEB0DC9A0EBE0645D3A12571B29242C874263FB4E"
02399     "6C6F55DF4A035C46FA37DE67444B427D52FA662D1B6BCF91E1B310CFAD91470652D72E955E"
02400     "5AD4CB2FE397E0AD9BF5E5874DA7A1FF38A6ED5763020EB20C2D29E618DA414958C88620D4"
02401     "03562FBBA7BB206FBE5DF23836692970FE8AA02E42BA387E42684D234CB2EF9369D8860CC2"
02402     "129153E29E3C27660AB6DBD9875A7D8B15DF0915A3ED7D1F8D4072E161441DB969664AD325"
```

```
02403     "D505A166DDE4A01674CE6E9509FFC6EC801AA9AEAD1A03B4F15A80C315871D8074783A72A5"
02404     "09F16EE18B75C2A1C8BBA2E7470D99916B75BF60BF0DCE946D9FAE5BBAA0F29A619C140F67"
02405     "859DBADA0B83EE8EDEBA8ABCAD485E1F1F7E1187370221C5E64955E6B0175292D2683344ED"
02406     "CA3F7818EE1D59B0F14E0A0D03FA2EC1D8AAFC986C5540304A92057E18CE6B1F3DE37A4B3A"
02407     "8C34778D6073A198BE62882C84DEBD29D7387DF3D0",  // PRN 17
02408     "3FD0B15E949ACD46F1FF5A6AF28CC0C85B149CE83F9D6F575BA8C612D3C219C37A4A880541"
02409     "239366839FC927A4B11BC6401ED48941F86BD00FAE5232BB44092786C63A3B676DE8BCA175"
02410     "B457BED1C099F83A6FAE755777A4E63F2AC0E369BCF0B428A5251C4B04FA7B3B94E88BC4B6"
02411     "B91797B0AD88814C72F10B1130F10DB4DAA00CE3ADCB0C5FBB1B3FCEAD129C4E75B3684A9D"
02412     "7B93D520C88D0261AA6DEE1FF5E67354DF2DC426B7E5106F613CF46D819398A856C9935CB1"
02413     "4111522A63CAA36430F69B37BCCEFF5A118EE51DCFE6D20C246580D02213F3C3DC75D4D260"
02414     "9CFE141D0E7210283BFC55C87B3CA52FB44D3F105C96B1AD32C8EA3B87F71A199AA19481DF"
02415     "4BE31A859E8A9D22DEC274EECB2874131E7A8CACC337343A969FE4A96EBB32007BAC4E80C6"
02416     "896AAD0D371E4C85B4CAA4B2FDB76716969C1275C6D8CB462E9E96DED97DE86122E9436130"
02417     "C171ED611BE4EE7B93043DC7262FAABD369F5F77ED35060F8F0A3A20917CC88F05E0A28305"
02418     "3F680A606154F0EA7ECE38940D5E78E9EC944D2385F44B00D3DD3195FE122F5173F32C49F1"
02419     "25930B0C9AC644B7415D3745802F8088064AE7804F8ADDC060AB6205CD16CDADDB6F697138"
02420     "60B378BFF0B7CD3565F82CED1F61D3C3386D83F35B951DDA4DAC06897B92F83ECD9D96A533"
02421     "B81854D609F71FDB2B743A25EC25C4B1BA135A9C3972DEFDD1F72ABC4D41196D48947584C7"
02422     "FA57B1D017E531F58934365B9C2C269452FFD3B99597239695A1AB154BAE6C152DF283DFD9"
02423     "61999392FA406777CB7824DE08F9ED5737FFC4ACF66FAA27E864B6F2D7BF5F3AA16748CE56"
02424     "274EF0D4612436EF0B8688DC6A6B77EB520CCD991EBFFDEC26438D1E4344F781AE0C27CE6D"
02425     "C466250D99C87961ACC102D9518F67C2E90B6B9B9640AD43C64570E24242DA73F060E8C460"
02426     "0BFE9FD27182732CA3AA8364F7872961C527946E06A392E1881BBC9D7875C1770ECADB8ACF"
02427     "1F90A6E3C944B26CA1CCF6509440301C60FE132B4DDD35ECE30ACAD661F85B633C283DC777"
02428     "CB9AFDBBBF2D95CE29BC5DF684C3CC9FF47E2B55D640B65E502D4C548738A142D4B0831C63"
02429     "1DD46749E617877415FAD96286A2332B677CC355D47843C00B788B820231CCB78942115589"
02430     "2B702CCABA429148AAF206BCCE6DD59488C23E30EDAC1AA2D5C046B7DF71AD9335C840D641"
02431     "8C548029D33515A3B1D9609298F80DE4ED2F2FC534CE95CC90D2143190C35DBEE0E6D83279"
02432     "581748B7D3BAF9575A6E62951FB90A866FBA956C306C8E34B4904682DBC7711A1B01568BC7"
02433     "9B55A18A7426F0988466EE147DDB723F3C225E39ADFA3E020D862EF334A26FE139B1D51C69"
02434     "D56F6C0D8F3A122B718BD49599403BC508BB03B5F424C6723401C4727EBBF0D06227AFA06A"
02435     "056726C35559998969067278DB89A99788450137728693F5304713E444E40CAEB1640F0EF5"
02436     "357B3E9D82900B929C8294EB2A880DC03C7E71A8CE761D64FD7C337053B2C6CDF42A4C49F8"
02437     "BDF1360192D539DB104BFEC4FA55DF6606E8A727F89549E286C189C0B7E003A6CBF7785940"
02438     "079DA124EC878E06D60A233F5507D8AE1B87C7CC5AD2CBADF5B8074F5A02DABFFF4FD6719D"
02439     "5FD5312E4724BF8F72E726D8B85DE43C71F4BBC009E2AF01086468C60A5BD48EDE5EB5A1C1"
02440     "307B19826D3B9D9126D0AFB69E04EBC828CF2BB33DCEEC107A016E1BEEE9898A2E4A6F82E3"
02441     "5641151730D1F74CBFBF2903FB6F8AABA099C38831106DD169E0DC567744D054743F4FA1FC"
02442     "EE05ECE00B46A453E4D3E8E39992266266A7338ED4",  // PRN 18
02443     "FCB7CFFD8ECD161BFEA5F40252E37463CEB8B0BC69606B5297306CAEA2D362DF5D3D9DB584"
02444     "15593CB15FDBBA7763EF6BE72FD34C983F0962341EB998F7304FAE9973E54AFA321EA23531"
02445     "D48069E9F594754CBB0CAE12B28AE611B646CD1DF5FD9C24064CEE1A2B69FA82845D01CA0B"
02446     "DFCB0F6630445BD7ADAE0AC8E89BD0FD927B78B26A97AEBAEE00F43EE232210635AD90A318"
02447     "AC2006DCA0A68B24E4A51FD4343DBAB51B1409EBE1D985D4032BB5AAA110C036ADC11B655D"
02448     "94F6F9B789CA92F0D9AA174E6FFB40E3538654F6E60A7BCA32338A136BF1D6F7FFE784F536"
02449     "91FF4623954A043D071D893BC25FB52FC86C82DBBC55C9DBFEA045D004E5F3ADD9FB1BF1CE"
02450     "B35FE0721473E7082CE73CECE4B3B3866B0D12FAF7E4A33D43CA7B0058A833F8BB7704C1DF"
02451     "303829E417B3889D61B66586396E23D4F9273DD7D388877FDD431E07BCA7364D051A3417A9"
02452     "D6FE3FC88F4F8394C4C7AABCD2C7FEF4C3F428434541A7CD845ADCE722D3155AE58924B5BB"
02453     "02CB64502BA7DD67F059E1D8891DA6045AD994DAF798954059CB46878CCCC52525F66B37AA"
02454     "7CBF4790136C0747306D54E83050C45AE5A903A6726B82930448ED4C0E2B4819416E2C01E0"
02455     "4F9D4905856D6F38FFC0BB58FE0090793E12AB8C5D769D17B928FF258E9D64C3004FC8F6D4"
02456     "FB4A9D8A7DCDB779554B88C0C51FB28A15F09FA2B50489FFD73270DBF273BD941C22018FFB"
02457     "B175021B21F9B43AF09364758B4CF90546AF1C20BBCA34064F674D46169F2665A0A28DD749"
02458     "9201E8336D3417A8264DC9DA7114C8366B4DADA1262072D9DBB3698CCED710582D4C26C00C"
02459     "0A868D3B4AC61C5F1FADA4E35368AF34796A85EF56B5945B8414EC3154A788CD3AC3A1E85F"
02460     "A10ECF1562283E7B1C514B3332972192AD74B423DC8D9010AD22C812FD6730C9B3ECB849C8"
02461     "468D98E06F093F2306858B8FE7B9459E9D1C151145B36ADED231B6FB4BE02F6A39B2362C0E"
02462     "9EA250A7A61752EDE8D91428996FFEA4D5B24E3359304A76A838F5459C60F61D56E5F8D767"
02463     "3BE38BF4775FFED272EAF58DA1367EB30AD696238680AB47B446C4E14F077F0A188A91CC05"
02464     "9BB7A6547C53317C3F7B59A0F61173FB8A1C517194BFE0E38E66EFC33514181D142BB1B99D"
02465     "5D651FAF51085433D5FCA310665B4535AD827A3313A99B8F2BEB5B1C0F2EF935CD9BB7194B"
02466     "8E8CDCA268FDFCA204ACEF4FB55B1FF51A17E28BC22F80E8BB7A49CECF3AC71D1DFA178A4F"
02467     "B497D03C777A7F24172B33558100336D7846E24FD65B38CC9B46344339DC55E76CA11061F1"
02468     "FC33C8E48DBB917E44175A1516372C3E665AE9647BFB5A6A643E0893ECF1606D4824CCC612"
02469     "833F5453ED49B82534AE739A8E5B07F20BB503632BD97FC7FBFC01E60C8DD5876867469FCF"
02470     "AE0D0B8F4A70EE4A063EFD074E46AC1167A0F694D5614BCD5035BEABBAC09532D84743C703"
02471     "BF01E46AF02A0A2FAF212C049DC295824F2C3C967C1094830BD8D52C8AB435C161154C98E5"
02472     "5B943DC25897C16FAC9011FF9427A4F605758B746C739159466EA410007B8602DDAEF6ADB0"
02473     "99A0FA09A75ABCADAB692C3746549D5525590CA0BD408482CCC7FE2DBA8451B1C189D769D4"
02474     "8BA6F78DE7017C6BAC99BC102ABFF75EA0E6D992BB5D7D2CFE0959081810D8A19346DD6D03"
02475     "61D0E8C5880A6D93B339709A3A57413E2E911B7DAC253DBFA850B9CB14C758B954A94A3560"
02476     "1991AD0882CC9D0D5E450010D3C9D782209DE09E03F167FD2D3A7999F6A0F9EDA2D960DD70"
02477     "6F95A2C873825C0F972B5B9D59A1A300235BCF4BC0",  // PRN 19
02478     "B83815123603B9E0F1EFD6323908E879487E0BD6380DC9B000307BE0E1C458FBDBA99BE430"
02479     "EC71AB789F07322DA6E83554419F27D6D9642FDBB139741B05210A3D1E5F62199C4FD09FBC"
02480     "31784E79AB305543C7786CF96B631825A3826FBA918579C4EBA81EF8378F1C299A17DC7B00"
02481     "7F289F279C0B2653E9B2FCA21B4E1E089263B2C6C4D1925374DF289FE66AD492C3F973A833"
02482     "A941A22DB23C9AEDCCB6B4D2B494A5F8871E03759137FC08B2A3B594B1CE25C6CC7D200369"
02483     "C303383EFD10B35D9E121FB231863B262F3B418FF4AE7ADDFFD040A9D03EFDED962C071F9C"
02484     "5E2BDD2A15B66450A23D05E76B1DC1BF9E8E8EB1023553B9920224E7D97222A771DD070820"
02485     "7733529F7B65F4F1975C7067235AE74C3EBBE551162F1EC737E05FFAED13A9AC8127E5C477"
02486     "435BF033F95978A9A2AD3A98E8D93BF180B1FB109864213DFCAC68DC9A4BCAF433751452A2"
02487     "ED7706A7FA75A8966778871044E1B0B016D5D07BDD707115BBF8B596D321064415268D07CA"
02488     "C56F85CEE97D906752E18DCF1E057B3EB7889217EBD34FB0E96A9B84B358FAA654360DCDE8"
02489     "00693D17DD12E1AE1F44A74999346A60F8FD52996DB2F0079973F09BFFC9D75F7A45AD0282"
```

```
02490     "E2AE969CD711C563C5C5257D6F5975E230504C308EDF8A1C094737AFA6FAA8542EC4009BB7"
02491     "88633312D8F5F1D1524FC0B909226241AFC8F0542FF20BC321A0C075AB473B7C3BCE7E9610"
02492     "5ECA0EAF0648F22F479449636F720E90B7BA442AAE5C2B9ACE8B22D3A5BB1D41A596560662"
02493     "229663C2D2925DCFE55ED0E8C561C44B9B6E20F96E0487E8579041AEF138CC7D4BD28F75BF"
02494     "D6FED2A9BF9B3D9937EB70C5EAEB363B3CCC823DC131E1D39B22939D4967376CB2BFBE454A"
02495     "A43DD6DF9526A5C76A822BF3ABEC03A50484F487567CAC163AC594485B49BCC3C553756C51"
02496     "5944662C39B60A45E40D83F7DB02627AA4299B3CEEE7574C28E8A0841FBE944259BCBADE3B"
02497     "5AF8B4B8493334F0E1B1276BA88F7069F7E94A2619F02B4EF1989D332C29E43D35346915E8"
02498     "09C310669D0AB066A29CFFB6ABAFCC46AFB42B9C01D27D13AF0E9473EF257EF51A4A073310"
02499     "15624E7BD6004D55C49CDCA19A3F36266E68D1AB2E780E755FEFE738A972BF6664A123AC49"
02500     "739AB0539C59AABDB220B866261700E7CDC14395B1017BC07777AE97F11568C6D640CB0EA0"
02501     "B7EC2DAFFA83477227D9F47C066FA889D4A711EBFAF5805D4D554963F36169F61D4823BE93"
02502     "A7A7BC15F0BC47EE9BC60EFEA91C1C5C4193E56D4488BF4716D82FC2BEDC3E88927DCA860F"
02503     "1EF9F1FF9F10145E308888C0A3218A31F2E108D3E7BCE5A083BCCBE41E0AB1BEBBC8D0F62B"
02504     "D598AB41A1878CB492CA88DFB3367C4B56B5DDFAFA05081C5EB020DFA5AB4ADE5B23B5649D"
02505     "DD23670C33F6A4D5865673F47CC34DF446DCACCAB896D67A690876AA7456BCF8AB2B1E2754"
02506     "29E9BFA732F80103685B544B836F0327A1B56928CD97108AA27A28B3E5D183B2A832DB045A"
02507     "A7254D811D10DE0CAC1C3515ADC4F5261794791B533EF51037671578894AAF5B8247E05686"
02508     "0147F32C214D1A219F0F4D1F93AC3EA7BDA10DEA49AF12D0E512852E959C36F934610080FC"
02509     "604E8A9A8F98C11A3E427DB7C2E3F887AE53F9624649F4B8AD27CDC81A0153B8E3C3CB4F6E"
02510     "333728379CD686E0A48736DAF27055024CED3544507C82DFB19208E9EAB93A32BDDC98D141"
02511     "E66B2EE5EA1895285BF27B1C78E609B6CD1A39AEA60F72F0580FD6ACEC63CC2E3A26A38C58"
02512     "3E0FC766CA3FD5DE377D3DF3F543E2DFC9AC399304",  // PRN 20
02513     "48224A8F4769215AC1C60CCD0D9E8C066F853A8CC4D8523A20119B120D5BC81F0D912D4960"
02514     "1A3A401CC6FA63574A49B56E7257F0C7B2F4506720A1003EA5EC06FA9C645690F05F859CBD"
02515     "B59928F80AEA94678FF9E3BF2C6924ED21EC2413C4BFAA853380F2D25144CABC10E122664B"
02516     "2678F206DE41A6DBEE4EF7A040B24833C5C30C6F892135BCEACB2D2F5F7DA30311FA19432C"
02517     "131AE40991D2EB77515B76F8FCC3F54379A1F040D6CDC597C2CE6C9580D96C47C09BBA89CA"
02518     "423E4A4D2B3F4BB3D9DE70DFC1D997C3ACFF77C60E413A502BBC8E6502E30ED755E187FCF4"
02519     "5CFC05961702953E1991072FBBCEB1F35F3E28AE1AD6798A26F48847A9CEE64025C18D436E"
02520     "691D64E71563E8F03AE38B1448C51CC7165F891BB83B9F1548AB092EB78B97FE6EA13BB569"
02521     "BC706E38F486B346E5E2F5289819DEAB1F4DB223A0B667A46F417C292FA52F80300F7315D3"
02522     "321993977E930C0C1A1C4A038B0CF97CAF67DD9DC45A0F1419C3795CC4A692C1E44FF4B014"
02523     "B3B36139C97AAA3FF5C8407E6984256516E7A51CA3FFFCAD008BA7FE05026D1C3160F70BF6"
02524     "104F9B9C2B6F5C38C68B83D7C74695923D3C5F8F71CF7C05A9E0587C728E9714FD8C11FFB7"
02525     "738B52BA12B3E1A760A05AD934B5AF81AC74DA28D045BACD90C394F844756237CEC12B4809"
02526     "D12B944734F94DB0CF6C171A5686C1EE0E3CADF64907CD884EA65FD4DB785F46EF4DBD6242"
02527     "789B42155548FA696BF18B7737FBBBDB28B98C79DDF1E7C4379221F339EBECC8552EB1F704"
02528     "DF1604C81619A4366F71753F8EF143B47161955A3E2413FE8A7C73024E401FCB54B64DE247"
02529     "82CB067F034F5695E0E5133AE7AF977D555A38DB3716A752C2BB292EC06048C76D64DD5ACE"
02530     "EFA5170C1AEB043CB317F47A75C8627A163079C8501B4E1D9260286C351299CE34041C056A"
02531     "DFF93EE07F8567E882AD3A9C8D2E7B990DBAF24ADD0E98ECD4708E5AA8F3CE1BF650FBD70E"
02532     "50B7D9B055E446B2A377830996902ABBBEF4EE72711DF1AECD618BB43B0CC8D4D3CA087F0C"
02533     "924EF25CA14650CEB514EF04E3BAAD9E7F1B7A6D0B821874C121D6EFB050C0BCDEB5995B34"
02534     "1810B9CF3FA482578068E5E835AF569290825DBE50A8B1A8F62E33757F0FFAB3C6F9C6B4E7"
02535     "ECFD93C98A529972A72A0F48559C16137F6E1DE08CE862BF76563586BB9DA6ED4F89DD0DD5"
02536     "549CE50C8C79792F4C68937100FF8DCE52BF335077EFF91A9674678496E90AE1DF2FDEBD54"
02537     "70E3A8A5B62A5034364FA37F42B19233DBA4F6EF1C95E4921E7E8AA8454FDC185071587B14"
02538     "135A1BAD13BF189636840535D75ED1EB7E7014C5044009E9CCA5A469675A24DC821591F8DC"
02539     "C40B78458AD792305AF2948CA3B262ACB75F78B119277D84CDDE94F790DAE6154461AD592E"
02540     "3F4BE67B36CA3928C824F02704C9A02122D7ECA027EEFD8F0DFB75BB7CEF11FD2240CC698B"
02541     "96616946DF9F513CF422FB5F811DA1EC68288CACCBBA85CE410D220B4F95B1D844EF7A83A2"
02542     "8F31249C4107CD76B77AF9012392BDD43D584C8DB4E10F6B997456B12ABFF5B85FB3188E97"
02543     "3FA8ACA99A532CDD93CF0D3F7B9B5E806FCECAE45671A25FA49794CEE3126F35F1F2B0AD05"
02544     "8948FDF4DCFBE08EAB3F4C05F1153F482033E762ECB7B4CAF82B487B421BCA7637DB621212"
02545     "CC13320B28755560ED11747F7FC0D205E599172686ACFB51BD21ED437012810217A6BA6548"
02546     "CBAB1526A438ABC0C66C1CBC422D821A4BB02F21F7063BDBAAA93E7343674204536942A0AE"
02547     "D09ED4F41E7A458F176A8BF399EE07DFAD47F0F6A0",  // PRN 21
02548     "0FEE25E2B7368DBF42023A84F771CC8BBBE9F4401605A100E3DC1BF165591FC7B19399B80E"
02549     "A19642D41629131B8BF913716AB4DAF40602FCC7501DC4D9F1F08CB1D27A78A688318380BC"
02550     "1826D38B61756FECA71D8FF5827DA53F5EA772C79E2794D7DC062CA34FBBE1221BECFDADA4"
02551     "61373DB6AE81438171B2BE9378600EAA562E189A3CCFD0C8E9B14A682EA2CD96217EFFE7B6"
02552     "6E7E291A5E3FA2BBE8E97B1EE5E98B42DB9A3C8AC14A511233A715225EAFB0A337BBBDB392"
02553     "E51787FC5CBE15E878B147D489F524DE451E4BBCB81BADEA4EEA74A31CF32871DE17FA4162"
02554     "8533210849706ACCDC500A39A7B33DF16FAD8049979139C1A13D34BA9C5FC38403042F67AA"
02555     "CC781B434E8F3293A6C71D0C165F2C168FDB2A4C5AB45576A8AE22FD09D2DF593B4D370908"
02556     "2153B99BAB2210E94416A88AFEEC18E0704B2D82D089039EBBC2BFDD7D837ECC8A5C731D95"
02557     "1E7C5183261AB947851F8250F8A13522567EDEBD580AFE9DE97B9EADA25B12A6EE8B7F49ED"
02558     "399CC4EE82EA2DC01FC7F5DD39A4B173CB0B97B01C872F1B1FEF6569FD5658C8E7DC2360FA"
02559     "C741F863262BFD7EAF1D39313161D31364A5E53CF8C803AE763B3B63F6C50FD52E03DC5E4C"
02560     "AA1E8D33CC97807872l4D79632D24D8A04D1660CBE1E9FC06194FD917FF3DF6F02871825A4"
02561     "B78B2D2C62D5545366FF7E6C656F0E4EF2A48328FE4A97D1E14A1EDDEE72102BE7516A1000"
02562     "3A5DE6776B57A0BB22EC2F235F16613B14409AFECAD4CB2DD16C4099DD70612A6F12FD17AC"
02563     "C89E147A5E0CC5EEBFDE24CE84A126D31F4ED7C66151ADB74E83ED6A6A4D6D3CE05FE92D45"
02564     "C63F3F36ACDF36211E0F9F2F997F28A0FC9BEC96E16BE28760E3CC366EFF2FE4F772D66901"
02565     "FAAFA022696012C801B8AA0F48217B051711E2D91A9380360BA93E6F40F1E9069772E63B37"
02566     "B44859ED791542AF04B51827003402DED69BDCADB886FBFB161773596F248A0F46166DC0D4"
02567     "FD3E7431BE49C8DD3940908BCE8D716EB6524B188A8E133E0F6D194920016B3E8146A773C4"
02568     "94427FA9D5390B143985F312004641D7300215E072D28D16F8FC97E14741536FDA9EC945B0"
02569     "D028A59C2F209CB0618240A3D68D230A03ECE99665D91D63E96C2B48CCED9232B1FCFA291A"
02570     "A28F101AF25874A7C3214F22CAE43DE61B9D1EF18451CFBAFF3DE7C864222E3BAE64BA5DEA"
02571     "C8E54A188E0BDBE485F576F506D2188AB1C839E8DDABCB707AA9C6036EC11E88C273C9AE2E"
02572     "6B8564B1387C18C61C8396E9A56E684875B7B526BF60BE71C38A7270547l3347D5CA464DE8"
02573     "1F9A5B8DFB8878B1E4EBC23BC88061643FF5052572CE5025093614449EB322ABEC46F2D948"
02574     "517649F7B92155FFD4928A8C094C22DEE253FD02413DE44A19C63DAAEC25C20F78D50E8AAE"
02575     "76DDA4DEBB7ABE2F6150D245D219D9AF2E64090ABD4546BFA781570F75C2D1C6E10EEC5763"
02576     "0BFF0EE6E554DA907288EDD0D014E8C79A8E647364BE0BE42842C327B97B20CBD63B3F13F3"
```

```
02577     "AF26929C2DAA2876D3913064856D2C382EEC42626F0672298ECDD2EA3E26EDB3BAA3F03D95"
02578     "DE26E22B1CD7A6D4ECB20D19A1311536329B6B5DBE8753C6D877B8BBAD5EFC386C083CC45A"
02579     "E43CC09E6559613079911AB5404464BC3024D5004646708F145BD5A714D8DEAD1EB3A91258"
02580     "BB5C79D99EE9917FFA7669F5B72E11AB4CA8C63CA2D22C2C7B54929B0A43908148FA5AB17E"
02581     "FE7047ADC92FBF78C6D66EA4FBFCA62EEB82A2863E428595AC4D7CA56B5DB79F210A18D79D"
02582     "BAF1A988191D2424624FA89E48A91AAD26D2A4998C",  // PRN 22
02583     "38D33B3C841911BACCABF31696C273DA36C219EE91A8F634B3EBC1A4AF8F09ED9AED1BBAE8"
02584     "EDC31CEC9E949D083073CB13F1D1F7ED660BB22EB967920A69736E5D31CB21BB504456A686"
02585     "3B625C518ECDA1435C12C5D7B2164EA34421B47CF476F9753E912CB401F8F6183F6A69ABF5"
02586     "D5CE152167419F092B77B9CDCB0E42C118C875D7D3605024058424CC8ACF77BEC2D0AE951C"
02587     "E39BD13DCE61069BE7950FDC0E3062BCEF8FF828662E29F7D34AF1D2BCB356459CA5F76E23"
02588     "EEB7646C1D125A95FCF55B22E306F110B01E5376CCB5EBD6A770AB7FBDF770844AE1AEA269"
02589     "AE3640ACB862F0DD15781992A13394EE22318A4309E1D7B87D8CBCBAA24F1D61F365935CCC"
02590     "F82FEE82C7FBE8B20B80A504EA01516D4AA7975923A6F177BF63C137B36910722124FC8AE2"
02591     "8B9C6D00A2A2EC8929B3880622455A94E01DCF8EBA376304D9AE1DBC4A5EE91D80F455016C"
02592     "F0B83D31CB75AF547CA4391C7366DC769C2B684AF3EB3520130FB48958EEA93CBAB12FA8E8"
02593     "91EFA7375504C7302B04EE822775066BEE720176DBCCC3BD5DCE8A1F2670B4345EEA4C40BF"
02594     "CC7A715A495EC755ADBE861C9DF7F55D2070C4C4C0503278AE04990A8734FDC630F373F7FE"
02595     "DBD929D252555A3572B66FFA254555573E3C370A7302D41A91245C4C7CCC4B5081F6D8435E"
02596     "F5F7FB56E51B2AB58FD0579FC4C66AE0C8B9D779C1554EA375D725C03983F2B4A16A419103"
02597     "58F0D5A8F205DC1179D62ABD7A8738DE8F4ED8197607623ADADC2F51575806E6F83C76DFC1"
02598     "ABED3DD4C6A9FDA817D92086FD6D1B03D4F4EFA997CB5B3572757FDAC41459BA087805525C"
02599     "2DD0803A18F887670BB898028F9B1657696AF9CEEB7E7498FC622059FB4C0BDFD377E2B2D0"
02600     "D11F37A998DFB482667BB9167A950E096AE1C15442DBA4C33A251D4762CE06DAB214789282"
02601     "079421B2EBE01F8EC29D5EF3268D8147D9DE7AAA6A117F8E9E084526C3394AB0E79E11DF79"
02602     "E83B472874BEC646853643EB0E140A4C81FBBB582E04E20209FAFE157AA745B28B3018BEE1"
02603     "0D5DB657196FB4215F677B1A787AFAF0D43B206E5F79D92B6A064EF738150DD4F2FEC7C942"
02604     "E5ECFF7830CFB8D67C8B289095BCC384CECEB2927099DD02D1358B20BA5DDB20919F2BB8D8"
02605     "2DBD308160B221E977F13928F1EEA454EE387C4C3BD69D6CAB26AF98F6820C3A51667ACB15"
02606     "DD6B9FF0A1A18898AC0E917007BE36CF6C2C4D6807110C9EEC784F0B955D782FB903433C3B"
02607     "BFAB9B1EC4B1493892E44AF2D32E240DA7D7F61F19C216918CD6EE85DF8F08D626D19D31FF"
02608     "F4A33B7CF61A060A34C8B720FB3355D3433BB5D612F68A25BCC03B44338F7905074640ED7C"
02609     "8D65799B90A693C71E1E67B1A2643EF1D36A280563AF37CD924BA3B16DA2C264172BD25049"
02610     "883F6EE7A06815AA41244D9707ACCF0921F0B2EB0AD901006D66D5CC02302A1732FD004B53"
02611     "32A9A919F71814A4E32E892B952985BC23D364ADFD6FD793F3E78F0680D206548F087BE008"
02612     "FD55BCF0AED3AE3C4C1158EA765FA9D9CBCD3E38316B85B439F4EF223FE87B9A70D09C6075"
02613     "89B00A70D11574DFC0757761EE2FB8A09775F2080036D442D1DE1A463231998E1FC11B53C6"
02614     "5BC071EE261CE62F1A919691AA7D87EDE03634F229793DEB5D2ADC167C51EF8447EC4004A3"
02615     "8DC49ED0BB6F86D976420F949FFBE908BE303E3DE826C3BA297D47E0096C88E6000FB26697"
02616     "D69BAFD0DE4726C5B744B5655C14FA21328C17A6CEBF911C381E5231AB5FA614387A9DC1BF"
02617     "D7A34E48EB09F8CF115816D808613DB91AB0D8B7B4",  // PRN 23
02618     "C135B97C53366AF456E70332D981E740CFC412A17F0037635EE03C7E33924D30FD3946E0CD"
02619     "6392960568E6A4A950C0F7E2A17A850A5EBADD6593927559B7E2F9AD23D290AFF8C8B41F1A"
02620     "AA026B0FB9C12B7433B1200109003E732D23E00495F4D2CDEF36B8C9B3EA186E67EC58D7D3"
02621     "AA1E295C1300D40D4216C46713949ADFA210932E7011680184519990C00A521117BEE87993C"
02622     "2AB4008D145FA3B32D65C35C6EF8BDEBA87A4580AF3FE899EC8435464D25AE9209ACA14DBE"
02623     "3AA78B1D042F6AD749F4B36E60873421B85FBD08D21FBF689557F2F68B1C32071DFB62139D"
02624     "8FB59B929EEAE7AD46B1C646461BA9D9A29F57154955D1118A0E03ECED7F2C7178FF43045A"
02625     "450B03793F7686ACDDF5A6FD2D9BA02D6D8DBBE5BA78A937DDBA95798419BBF88567DA242F"
02626     "5184128223E47D5CE8C2BFA794299BDC057940AC16F43A6AA451BA06AB6AD587BFA1FB3CB3"
02627     "BF35BC222580570FCD4756DCBF62DE781F5D6A37D51745C1EEE4BC7D40BFB73BD254B364B1"
02628     "AAED78A9C563B1D51D36F7CB70A61D48D11E7D8E3DCD23B7F22A8D3A76B8EF01DF43D0ADFE"
02629     "46EFA695BB1F649EE783BE180493DF05FB78B4BE57121455C30301EA4C0AA76F9A48C0FD59"
02630     "79F269408E56E60190B58173FCA32F5492848DC219A3BCEF81B0CEF356A2C6AA71BE3AE5CA"
02631     "0BE534380C0A1398CB89528DC5BD0AE60267877F8FE9D8F91A2E3B0102C31C491F1D95B078"
02632     "C0B564E163856240775D1A936263D0F6E706788E81D76057707894348202 1D7877536F480F"
02633     "A92A69300F992477BF1FB5702A2ECDA84EDCF444BD2E303E6760C09BC5677F90A764410C9B"
02634     "240D455629B535F187089CA5FB6FAB7C00A6B83F9D0FAC34728E1AB368511A7D949A956EB7"
02635     "86E2B00C89C78177A876FA75272026BFBB908AD4497C8548A7375E0D61E6DCA6265F9C72EC"
02636     "43B9433144B9A8A8F73A2DC5807A5687AFA8601D384349538A79920E7A6C0C421B8DD9F02B"
02637     "F83CF91215CB2AB27A2C49095F47B870F30BCDAA5675EC717AAF7130147FC062FAF83868C6"
02638     "B39F9459625ACDCBB81DFB161D40ADD3CA4FED01B1295B0710CC35DD49909856422CBF400F"
02639     "ED086B71E54CF882F32DE69590F3FBF3F220EA567086D56DA3CDBBFEA7EEC10B6B7FEF02B5"
02640     "3D6B935D64CC0ED7CA1387767B522D06EEB39ABDDE8BC08B3EAB9B7D3777595DBB2BFE0B6B"
02641     "ED5D4D60E7CFDC399646BA01018B69FF211D7B39378AE25504FD1B189B218F4D8F5A358BB0"
02642     "B65746C362F4AACC38816029A8FBD4E8792ACC0A0BBF4C4C41FCF3ADAF0DDCEF973D76C68A"
02643     "8CB95AF159956ED1B8DA0BEB9850F025101AA521A99B1953C29CA073A3ADFC98C3DAF757E1"
02644     "98E428440670D8D4393C0D215E7872E9380637D45717CF349BC4DBE6B97BFA87C55B1D3124"
02645     "AEBF3FBB9663DE262FB93B3F84CA412CCC88EABBE4589ECA72A76D2C314D0B2959C043AC6C"
02646     "41DD3E8D0EC8C016B4338781BDB60A509A992A3073842F62BC4492934053B70216E7B74936"
02647     "A87821F261C16690BEDCDFCB0581C00D4048E60FD52E9187DFBF7867DBDECA23E407D032E9"
02648     "AAE409F2F52992C12810AA7DD9D4F5DBD66B89415861F8D892AA09B00DBD6C79C04741401E"
02649     "F8CAAA6DB0979430615B6E1B06788D26EFF874041EB3E47940B8DE8C11B97A5C0C29B8DB70"
02650     "A8F77F5937F6DCE3F07CF7855957CD1AC4A8F82C7C37DC4DC76FF4AF5DB651083C207B1C39"
02651     "4678BBF3B4547A047BC317B65FFDB1E30848183DF1699FFC67CE960FA4892C96A1F678BA38"
02652     "9A7D75215B69B4728043C220283378C184B3934418",  // PRN 24
02653     "71DE13302770E3EBE33872AF8677F9D48BA4C57647353AEE28349EA576576330D3A5C0CEB9"
02654     "2A9FBE09CB8F56F87F53765BB1CF0F04E3DC01FD450FB9832C3DA6ACFA0AFA72B1806F6B4B"
02655     "6D5EC9EAECDAE0847C919D49491EC69AA8F453D7699CF944AA787038782BC3A78628611A1E"
02656     "FC22BCE7711795CA62959CF9E89C12E64EC57485EDFB683010B8C510797614AEC2AC8F28D7"
02657     "0358A94B0209A9706487E6684D6AEDCCFE7744D291A4D08DF9B8D49ABE9CCD8209D080F075"
02658     "DA64F36BC847C53CCAE65ADA23DDBB51A9A701F225954612BFA5D7BE8933A9C0888E158DCB"
02659     "BB660C954BBF780470A2C7BE86C0E398BB260591A0757CD592DB7F2DD9BBF535BA90F82BF8"
02660     "505B4F40CCAB0E75B82F78D10A68D75BF49A343713EC025A32ED56EEFFED53ACCC92B1531B"
02661     "F9F0B34028B49B9B5B6D627274401DEABB44E3C226BEC94729DF2DDFE7EE96595217B6E54A"
02662     "CD596262AF96201DBF99BCA8E967990644044 67B499EEF821E958C1EED0FC87F00B7508097"
02663     "970C27FAA1803E9CDE0B727FEAA75838093153BF87B8F915FFB0B5ADC3598E426545AF8581"
```

```
02664     "400C26584F9B9D32128C22E5C94181561C3BE6770B662EFD617D3E58A9239D19A1E7233867"
02665     "F5843E428893BC662F87258AA8155B5FAECFAF71D295A06A4EC969A178B5A675E0AA6A2A53"
02666     "25B823A8D6CCF5CB96B818FD48ED04EE92941F042E7EAC89C5C30EEC06BAF16C7E4127D975"
02667     "10FABC1F5052BD3E6E3555F1EE48BEF8BD2BB0F36EFC827DC06CBF18BD3B1EA0A878090079"
02668     "E1200087ADCE4558A8FCCA3CBEE14B8E7CB99CAEA5D5D0797DD050D22F0467D6F7203ED323"
02669     "6EC0780542D28D01382D761FB8E3FBF1A8EBCB38AF1127787D2B35BB834124116BC2AEEFC8"
02670     "C5F24833D3ED8B119BE314681058CB197A11B8220D13D6A8267E4BEF19EBF3689EEAD02FF4"
02671     "57543CAB0F8B973B0AA280BBEAB79397BF58196B778B3B389622C4A60149F9F03773714D5D"
02672     "A8FCE40BF1A0A207B2139741937DAAA0958DAE6B4BEFA8C77285F0B0D60F73222A1558FAE2"
02673     "F358377AE744A98A8F382FBC80290D3FDE39F832D4C211C26BB33B78E538AEC6E309FF040C"
02674     "130DEA5C1B67E5292372B23368A00C3CF822320AFB3D1D84A278CE717AC362197A88D7CFFB"
02675     "CB8A86794F64532895EA1DD5864EE2BED85F1A25552F000D4DE6DE2CF7E9E064503C027C48"
02676     "0B6AFB98C819AEF8745D9EFC762B2F15177A1D50597737F34E23D6AAC9A4A63138E004D3CED"
02677     "605622AB0B4C8CACE0D26FA5D96ECD32027C9C3255DD3FE64FB9D60AA01AC7D3A03E78C65C"
02678     "5344E3449668AB45CAEC2EBC9210191AB5B1CF0CE584302401F398FCDC38339D453512A042"
02679     "3F03F07C6602759BC71894BD375B3E20CE8C4D0EA065753526F7785E56CCB01F87284F7C1A"
02680     "1CD9B65C60C7387AF6BFC83311F541FDD16E8FB02792DC32C61BB6C30A237952C9C3DE803F"
02681     "D53BE636CE67EDC603A80225ABEF84A46A07176B5ECFF2501739F19C4840E447EDD969825A"
02682     "4FEA6EDD58BE9FF25E5A54C53B47C115099C95C9768EB4443B6106FD134C283086D6C250BB"
02683     "185BDBDF0344A30F3A5BA65294CC9D17143A22B01F84C928903FE06F21193FD336411F70D0"
02684     "69217BC262C643E09B75BF287C9BA8916CB7A4BC0101F913015924A50291DC0125A9BCF866"
02685     "9A167575732C792C7198DE3C8FB1E26648296862544F18B59B1C0C1C2376628D5E6587E0FC"
02686     "51AA50567C1785F17C19C61E4FB7BA8DFE514E624D0044F3D5AFDDF38E93C6FDFF9C4D3C53"
02687     "1D4C8AA7F93917464B1EB15B3B90386D486ECA8098",  // PRN 25
02688     "7E8CFB74E2DD4318FF211564D342AEA2EAF031332F27616E5FB202E50B86580C1CE5B559AE"
02689     "7392EA24D52D446A28B63B8A0E910618B70E9E3F1CE264A10B180E27DAA8BA5DDD80B63081"
02690     "6D971E7827971FA7A4B6F55496A908602CDB20DFDFB5CB74BDB251E91E2E00BB1B2D6C130A"
02691     "EEDC46E73CA2C15D0E2F37BA9B4DFF057C3CB4DD0767F1160B833008ABC8CE46B879FC6DD4"
02692     "7EFDF5E6636B0BC78CA7C80C883F0E4475AFA5E9290956A44FEA050CBE08231C9E75EDC9FD"
02693     "E44A3D50DDD618BB97B43978179546FF522BEB0D9561E431B91B6E7F215FC2762D98CFA3E2"
02694     "CB6778717ACCDA4C19E2A2A5C9526AE4272BCD140C25093EA43FB99937DD21EB4068F6E53B"
02695     "06B105061709804F5D385279D4375522F1707637E42B4E748F73C90DBEB11986BAD53F0D15"
02696     "27BFBD61E6DAF12ABFC0B962CE4E24EEA180141A62FF062186E27130B7EF1AFE18F3D52BD3"
02697     "60A4A872EA197952DAD78C1A59318349552DE767DB1FD532769F1DEA34CF68D6E7B9B71D5C"
02698     "8F8C241BAC6E619AC5C8C41CAFD804209E7CF1A553C3799A3A57621194925F15F46FEF94FD"
02699     "9E308DF3F3692C948E341C4F65BCC8E684342071D1342630ABE91911AC4DBDDF0A012A9E41"
02700     "3FA22D3F51EEACCCADE6EA1444539071783ECE4E492E14A2F2B7FCEDDDEEC90F5EB6D1F5F4"
02701     "671A6C038D75508833B608A7CFDB3973EC7AB007624F197828D87F6E51239D000AFFEDBD57"
02702     "0E0DD1AD3E9D76AD08C751A15DF02D6E4A78124344C770A7090EDBFCF069012321FB146DFA"
02703     "1B069798B2EF74B7A206511BA29B05979094387EAFFE56B804A1DD0181D177BA26E4837C28"
02704     "64DDA836BB917536242B19BB9F2E9C59D40C25563DE3B84267BD7F58229461029A3F39C681"
02705     "1643954CCDC47F72295B045E3A8BF6E0EF367287BEB38F4A7F6BD5B6A90FDD1353D0415F4F"
02706     "51A8B40989F75E49E21E066DDD85097E350B9D01F7FB7FBF0AE119AABAEA7F0F5EDC196E92"
02707     "C1D7EEEE62E8CC85CCD466DF2D865A6BD7F6BA0A109FDA161CC61FC9891E26A28B4C39032A"
02708     "ED9ECE21D5071BCADC7592AE3B207580C5C39C94A8F3205140F46BBF3A3A9613678BB05714"
02709     "D13F687FE59A277F4A982968556789C5D777E01699B970C6C9E0F7E18E99026453F090977A"
02710     "38ED7CB8936FA930122BE704D61DD322772FF39E92423601447 16C0F8D76E46E189311BC21"
02711     "751F3E32AAF16E5F1FA14CA5E7D7B0F9D211EE06375BFFB361F6D8D2D9D93BF154B50EE5C2"
02712     "CDC7455E4EA73E2AE03AA086852B6373FC8EDA53854BBCD27CA7DEE6723F7792F4327ED5D9"
02713     "C942642F92F98943F7F53294613210A39B149BE7A8835087B15A03D3866E1846BD403FC503"
02714     "8B86F95664226A6BEB11FB4C6FB660A234297ACC3FDB0D6530359C03888BF8122C84319577"
02715     "044939E5A6AD4FCDFE16E96E531ABB021DBB9058C1EF1D85A21F729E2C274AC8A3E95084CA"
02716     "1CF4BE892DDEF2F7BE1FC35B18798BB4B320FCD65CF8A3E32C0ABAEB84ABEB71EE54C98905"
02717     "2D1A640E9D04714255791788FA74C7D791E0F645DE40679E89E0C40E00D17842AFFBFA2803"
02718     "C1BA63C7734215F0D796082D7AE03E86A5109AE218CD3405CB6E8A0FD85281C84F85648C7E"
02719     "110D4CBD4D7B5EDFA5B47C2BE9F1D760487B47F51B67B6A7528FF31E9741D1E8B8B47B85DC"
02720     "CED466EFE1458CBF45D7E189A88304AE7FB10CCBAC49905AA83452027161A728B8BAFE3B8F"
02721     "C276D04A957519564ADC181B36D93F31227C6ED08177E37724450499D07A53A61484357CC9"
02722     "876353510F62707AD967C890C45B2F697C36BB76A8",  // PRN 26
02723     "B536C31B1447B59AC3D787D47D1B9A2402D7DD4C36CC8ED1530625544D93EF0B31F3516759"
02724     "E663FF7CDF38D275B437D7E59B222223930BAAE4CDC9D43680E9B18190CF794EC4B202458D"
02725     "10673DF57FDF3B52953AD67CB496E7BC09BDBE766A06170CE220A4B43FE828FDCF0BD59093"
02726     "E199E0E0EE354528863001C537503FBADDB4C8C2AA72F4251249C7063074B895D8B96516F6"
02727     "495405A6C7E1FAFB8A302F50D29920C87FD91A240C90496DB12653B38B6EC194960DD172DA"
02728     "007308DEF07D4C86054EE06E68C118447A4BE5816754206B5D04815B4742CB1E1EDBFE436C"
02729     "064523756F8C1A2D21BB02CDBFE223F607481C85CEAA3B1E3FAD53CC7969F19D4A53FC7C89"
02730     "F31467BBA88ED29909231A29469777230E72B6A11B3EC0A1675C635EC298E2613FD4532350"
02731     "FA5BC9E13C6036C42911306A35D24249DAF09E88772B28DE291EBC6149617D220385956796"
02732     "B2C4EB4B4CBBD1B2958B10EFD2CFB8D8A6363C1BCCF4EC5287928F948F1F4BA3FC8A0429EB"
02733     "F34294E8CC56BB9A703EEB1C02E45F8E757C4391671F6AAA567EE5D7AC58D3F8C9C4C788D2"
02734     "AF0958F4D1FBFF1305B98254E43BC88DEBCAFCAB36CAC4FA92F20BBA296DC940D8D49F88E7"
02735     "303696DEE672EB73670168D733A16BA0ABCD2252791742050DA5B023E97F6F6BA5F3EC988F"
02736     "DE476CDD14A48CFFE8134486F6721FECB549F4F4991D6A56445B06E81BBC1D68840CDF38B6"
02737     "EB2CADF760F08130BFFE6C695C3EBB47800B7E1EA490E64F52C3192EFB45C65959B40AC06C"
02738     "BEE80ECD0C5EF333ABAFF3E63DF46021E35D1FE93341AA0974E298869EB27736020EBA9F79"
02739     "B4CF49C4D4AE38526D5A100AF7A7AD708D53CC40C6D520A56E98F94F84185564FB17EC3E22"
02740     "E4DB6D4D873090E47B72D12539A4EE87A7C39A50579B64A2027CBD2F147E4FC89277A90340"
02741     "88A60129790A4AC0764D28F91D8718F52C7B1817D5962D58CA130D76519968B1189CD14986"
02742     "FDB592C05B14953AACCDA46E0F7824095C1FCFD8F4BE25EC9F2B771D813B2315F040922CF4"
02743     "D97349F8E5A2E5DFD100A7010E3AB6B5065B6357F9DB71CB255AD202B3EB364501E51CB555"
02744     "0ED59F893D5DC66DACA85D8BA60FF733B200026110A1070264C3A4201761808E7A9F998CEC"
02745     "F1AE00ED4935E7D1BD7DD4155AC6B87BDF01A94B19444EEB20AB318377050C513E98A611F2"
02746     "238955C9BAB3231D8C7A0EA2D20B206963B8F689B08D4B21269FCDC8E2AF83FE6AF0AF200B"
02747     "86FD3CE1D6FD870C03422E09961349761E8AF6C18013CAE65773A2A4E6262E497FE2D9C204"
02748     "A6D0151075638040AB48DFAE0801D032A61AC98C4C9B7DB80BF9068C3241EF2CCD352CB09C"
02749     "DAF391C05D6441CC32D91DC9ED7A93677684B150DD1EB0499BE2014CF2CDA7D264E2F5C7D5"
02750     "37C3DF97E32D92B52224C1B3ED952C58F1E2531965E64C642417DC7D6AB91B3E6BA2E95DA8"
```

```
02751     "04314E10D29E03026F1AEF4ABDFC7D1FD14B9CACB79B3805A1CE5A7D83FD1E33C7FD9337E1"
02752     "D5BE548DB9BB37813D83832BD1B7F5D612C38B18C196913DADA7C1B6FFEA6A1169A0197657"
02753     "E4E2EDB02301A834B362C927AEE79FB4AB178E2639CA8DFDB1C7D8F710CFE7EC94286E8122"
02754     "7A3F8B6DCB7432273F9421EAA6D6A833D4DBB32B19DCC3FE505FE9B9494B2CB18474BFC7F2"
02755     "A59504B515D63B50A7E259F2F1570B4961DAC9CB30E122A8CF4D3303FA04F610657850111E"
02756     "CEEC6B7DFD14E75C4FC65F4C331DE1B7FBCF6BF314E5DFFD974861202784CD4D9054C71434"
02757     "D1DBA42C3154743562B029FE03D6906A4C26B34CA4",  // PRN 27
02758     "B8E68CC3EA746245BDCEEF9B1F97B14A0C8DF0844B23A53379B6317F61ACD608E92C7A0111"
02759     "902BF515689A85149B20349AD7B6EDA5B539CF8C27D12FD6C7F30111D9251A5682E2463A52"
02760     "A82DF0C82CB45CEA6585F3960ED56BEC8415F3056D340987AE2F019B370338C3A3F48DAB12"
02761     "9EDAF018D16F6F22E003942B2A6747577B7549F05C369B76F2C88E13D20CB5BA1601D05AAB"
02762     "5140E8D95DAAC8C419CDB7FDACF3ED0D6FC9EA5D7ADF811FDA3A08E52315E46124212FDA67"
02763     "7623C8C3E999270EB086DFB0200D2256AA2D63E2A649BA2FEB0CABDAC780BBAD5E0B1323FE"
02764     "2205E4D64C090449608C2719157F4A56EB5C6A90FDCFD054FAD62CC2DCC559ABDAF2D93CCD"
02765     "F0EA623A8570F984A2F3457DE06F24D4C178B47947B57B698E5DD62B4D45CFC22DC5A41B9D"
02766     "B4AD28F4184B9E99D64EC9F0962F5E8662C67208EFA08C1EAF6DF12418A1E39B3A6E829BE1"
02767     "9122DCFDB49407C054690BCB2FB8189E4C005B7DA59A15CFE6744FE53B1E204FA28B5E515C"
02768     "AE94958CA43F03BC10E6B8138E95CA97AF0715C15F4108D9FC70A0F20B57C4E3F5F3C486D2"
02769     "79D6A54D643402D691CBB471B3318864D46F57C11B86DBD3E5316482A6125C8FDD52706D10"
02770     "93917106DE1299DE28FF4C4EAFC848EC623BA828A3E030BF0C1883805190D8213DB19FCAC2"
02771     "E0F50A54F3522819B4CA4275FAE720EC6454129192DEA014BCEFBA33F487731AFF9CDCA59A"
02772     "B8B84B2FDB302E1D6F49F7ECF828E2C974FBE023FFBDAD926068644DF44DD50A1C4F341EA7"
02773     "D20F9C8223678CCAC94F756D33CAA620FB14CB9AF5E20A2CB70646F8047AE5D4116C6B8D1B"
02774     "E67DF616F9585563A3485B06D8EAB51841919DEB3792C5949E9A6CA65B34F13751E1EA6012"
02775     "2A4A869C28E2C133AB88FFF40ADE0DA21040D580EF84D6BF683D5CD53D6D676ECF5E290763"
02776     "9E94E023554E9065B9CCCCD0A451FEDA50F799879BBE559C324F7E8EE2E7A8829DFE94B57A"
02777     "38A2402BFEDCD6C559CA7EBFA7686E0232E9D68EAB024C37909916567D1139FF9B1EF81972"
02778     "B2F478076300B57CE5CBFA72752915F5AA97BE0E3D58CE1ECAB7638D71CEFA25E822FA65A8"
02779     "E21F704DD24FF68225F21B2C253C06366652DC73278C95849C6810948AA2DE221CBC803A3A"
02780     "42FE5C58E223EB2069B460AE43325FABA14DBE2442E2E3C7F75D1107ED802BE7BA96CDA8D3"
02781     "02EA07F8A5BC8E74A8F4DC88E67DCD197DC47DCA3B1FDD55D5E0FD7D90E2450F77EE89505F"
02782     "8D3934B8F2F737E06B1586F1133B7BC5A99D578FAFA915C2B50C6D90F4209732A5960BA585"
02783     "84AA09B4A2ABB01AD43D99A4B7698FDC72FA23AAA7A5DF52DAD57400650E5CBD003E29297B"
02784     "5076CA5764A1CAB66B8EB2F3738E1FD8F6147CD8785AD75C57068A4D483C1FB6355988D8FB"
02785     "821B6C1160658016E128D0EE2B1810FB1C72C46AD7227DEFC52422FC8952B9343381A25645"
02786     "52F598D02748C435A201668B91F2D0F55518CE19CA52C94DC0C813030E7BCFC6EEBBC4B185"
02787     "D6B8127703849A6C7BCB0D20A04983F2B54F3CDE7D3EDB8CB14961318462A3F8EBB5B17915"
02788     "80FEE399AB43A74743979B825F76A578E2A0D43223725B983E66D4E9C93ED3AC2FAAF0CFDC"
02789     "DD06CA84777144CD8D9181DFE96ABDC966A232A39D80157C94EA009E2694E8DAF4820C6FED"
02790     "6BA9D33F14EF6A35432CAB30F241C1EB10614C9765C33B9DB923E3C00610FC24EBD65B98A3"
02791     "C35F0D77BA3FC3255AA0458740F21C8A2CC6CA17C51F728C79B21D4C2F1BBAE25289480F4A"
02792     "D7BB843C454FC4268B87CA790C39C82DECE43BBC9C",  // PRN 28
02793     "1E7272D1251A8B5484E3ABC21A85B0316C8B3A5A1B25EB8FEFEC5E867F6DED4807D1C476C5"
02794     "B1506D50D9D3149425713A11E798F5415D11A7C48746D806C59EABB56240CF0C997CC9554B"
02795     "3FE3F20F785095DABC5A5B509B23BCAF52BE692D76D92362C524148E884239B085DE22B756"
02796     "C01DF9C514035E6F5295189B2EC033A3C65E203C1DC7ED072A6400FFE8E626A707A397B7DC"
02797     "8D27BF24DA0FD9DC2C3879F0EF90B0B8477F3B2C315F8AB8FBB195F7605FE749FE5DBB9183"
02798     "964D8CEE8B1A885707A0EA17533B84A17DED1594BCFEE87DDDD1BFAB10A802F818F9EF3036"
02799     "0D57F4D1F233653C8C78D8031F9D1B05596018608B3DA825F4A55F9753E037735F22A8F4ED"
02800     "26CDDCF8A77E82B1DB19A207606341EE965BAA0D8ADA95390C71ACE1366FBE47B84714C9E5"
02801     "3DE020108C424B1A961A4656582A5BC95AB14EE6B7648C01A70283DC287FFE282045787B25"
02802     "2B4FCC2B174803BE7B57B62991E69222C5525BF6436BF092BB93DA680FFBE1BAF7ECB41CD8"
02803     "6EEAA5BA76AF45D24447825BA5E4CCA2F03758F06910384212BB48775F44CA9F0291F2DFA4"
02804     "CE5DACB1B9E9A493723F1E1499EAF3B1248A893C2193C9AEBFECA9A8BB3A2C0D096BC181FB"
02805     "64E4091CBF9F36514024C05C13032B8FC7CDFB09B3178BD29F6EFDDCCAA74803B82C5B5675"
02806     "5A681FD5A28DBECFFFD965579077DCF0284CF9FD8D0F94AC4416B3BC8CD0A7C3C77F0ED555"
02807     "D9D56DAF8A6C1816A4C599718E8061AFFADE0D5BA7ED4313FFA27B8C4CADE6773D715F23E4"
02808     "3530EFF6B4F3751A1DFD22703D608EBF74F872D0BB7F84C74FF7095B19EF2555A03AF262D7"
02809     "4F43A33B645A9A716BE4A9CB5FCBC6C62845166EEFA521D18AA98AEDCC711501E0587EB75E"
02810     "78D6A5C82B60E50F4D961791D7C23A595D3B949E94CD8EE844A71FA12E9B58E23AA9F68139"
02811     "8D4C43298ECD06EECBD3B71A5CDB52DF5A3C9649DB486256EBDD631CE07DC56A0826DF7082"
02812     "E77165181F555A5C03F0F8815F79A8AE16FBAC166D13D8268F6EF844CE5E6D2542F0C436BB"
02813     "C85AE8E47723DF6FB90A36F5D7FA161CE15584CA32C3F1C8C4573C5DB5F0F5F362A063E94D"
02814     "99983D93E92646E2694F8A37F1C42664232A42706C7F3BA9CF7F5691B1FD38B0D03364C55C"
02815     "69A1C6EB9367EF1C77E9F35C5FA898BD74E7A0ED3F8E68F85D724C93DD876D9EA074E0DADC"
02816     "F57834F6210897D530049BEE77504F13DDA44D5617CFE96E8DC4746DA10A8DEE3077991D96"
02817     "F81393E29ACC32FDEE595F8BC18A251A6A343CA5A20D3D8BE51D6D6F7540DC4D2F51615192"
02818     "5183B7A05F58729A622ABB4A4553CFB12F99D4E1CF615B2F14E627BDDD4B15B7EA2956FE85"
02819     "36DE3510574FC70D4BA8901A43AF830BA7400CFFD614AD99A81B42B4C5D1A1BDC0C538A07D"
02820     "4B2431F971DF0D8F38B28B504C40694B001FC4302C35FDB95B09F589090BA32E5D883A6051"
02821     "911684B5613E5AE0360F71130D1853641C710DDC0C9DC48D034826B4F115A2B511CAA1AD08"
02822     "5FEDD0E739CB7306A1405E276A1473297B6CE33B0EFFBA37483489D87D94ADE0F71EE2869C"
02823     "A699F9D17871DAE5DD21265EF1F6CA340A4EEFDFF1D44B04EAF59B45509CAA51869367F99D"
02824     "61C5EFCBB74503B414D3712D79D490E65059FC78011EF17C73A1147B1F22C62ADF4978420B"
02825     "8AC0D6A977525612CC7A6062050D6B3448A28E93B11245520E9138F9E28337EAB428032082"
02826     "619C82C63A79D1504651ED6F4982E5BDB36078337D9BAA383D2A89E080EF319FC9C53B8BD8"
02827     "14B0577A5F7CF4B2E17B9B804045C8889693F09588",  // PRN 29
02828     "B75B69BFAF2260C0D84908BB648C5426C58339850615517995 93D3830E45449119E8BC4DBC"
02829     "8D68EA696079D306921D78F9DF7CE52A4ECEC6D5807B42BA30ED16F56472522708F3F333FC"
02830     "C8CBB3194535A84F84989191C34A80EBF43A1B677233E2C2EF63C7E98AC09E2CBE40559B34"
02831     "C0127DF2AD7F46005C4C958ED3B051FEF0D64B6DF3C5D210B53AF1A0C7A378E3AEA3D244A4"
02832     "7F8911D710C943AECCC2D3431AB672D878672045797CBD92AB2CB1E2D8DCFDE6667B230AAB"
02833     "0E007B8B7F58225E809E91A648173FFAB869EBFA522DCA8C27848D3E7D15E8006A1691AD3C"
02834     "846959BF14A66BA8B439B21FC76B6EFEE3DC15504DEA6F91996356C9365186A1A40B387C96"
02835     "FB7B15439850CF2A1308E0C9822D1A33D8D16AA56511FE1C85E23DD47E11B061B432A2C60D"
02836     "A7BC5C6C0885F7317C76DA27B1123C79EC220ECE197F89E46135CD677C3E4D6F32B3C4570D"
02837     "5C0F03CE18A8DFCE62E926B6A47833D15AEBE1BD891718F8E2B854EF5AEFAE7D01B0113B96"
```

```
02838     "A99F59C2D17ECB18A4075AF0E9132AF66F1DCE3EE877B12644C22EB4CFFD25ABE8D43033BF"
02839     "3C7F26A84ACC6F31802446C8538155986A9B233DE081D7DDB0989B463521D1ABB4CB4EBC5F"
02840     "C8CF9F519E253FAA5FE2BAA7FCC006862B9C7E6472DCCE6B90D318A08FF14A86DDABBC760E"
02841     "418D731CAC44BF82C92D2FBBB595209FC1DE600D49C9D8015955ACC08433F1D1CE4242B65D"
02842     "13DDE2547D6541B10EA8F58690978512AD967CD1B6584ABD9F44385D252C12D2DC6C3D2429"
02843     "AADBE8869B1B17051A16673171A6FC02B75E9DCD0FA7B4E1113F84B165FF0C771C75F3C72E"
02844     "98550AB0C75807DE98ADCF8CADB475D387884FD64DA73FDEA46073E381129E6B35571A09E8"
02845     "3D4970DF85AAFD0CEEAA9AE0C17B3DA2942213DCCEC15E243C84ACE9B85A5DB20AB1B89751"
02846     "9751EF3BCA451E1AAFA314D6D4FDB665CF6F3B7DC6CAEC944DB1DE75B8D2AA8BDC934A7650"
02847     "F355CD109F2A9AEC264F3DE2EA03A3448224A66894F2987D84FF8F3DD37E5B8BF30B6182D9"
02848     "8A640FFDDEFD034D1D8F43FE6811E92FB52A2753C5C52C58D74648C5A53E9C5BC898703E6A"
02849     "A4FAC21D0D5A8014B64A70F9A9F86500BC84B678B139F745025289A21DC1E0BEE6B349B8D7"
02850     "A98A89FD78DD2F30FAE4C45893B392DC8569924E04C102325A7980DDA8C6A53A14950239BB"
02851     "739855363BD3686206BD6456C5DE09D77745054B303B5C03C86DF7888FF2E00921986BA476"
02852     "48CE6F5C6EB8CE06E95551DE6875E18959B918CC3B33ECC4BE43833835777A39584C97B8C1"
02853     "59CDDDE9F88E3E4B5ED46FD68EC8EF6F4E0B7301D542E7E3B59E5EF5825D822C97CE3B30CA"
02854     "44DE46B5DD3D5BD65CD085FFF9B9707B13C2C7CE088ABE5E037EEBCF6384CE11A1243ACA15"
02855     "F443EA201C43FFC6DD55A4AF601C799A61D81CF00F3CC73723DD818DB17738E18ED1283C13"
02856     "FD4CA6F92B680D1A73B1BCB534DA1A059998AED115058EB87D60DA12CF15F0082A2F35277D"
02857     "012D1B2120194E7C94BA141D900B36F35D03BF5FB80FD3CD5F377F9BD0DC91DFB23C879257"
02858     "03A9F9551AB786C498A9BD0A4806F6B13D74331A1E45DB36E330945DCF97242E90FCBFFFED"
02859     "CD8D50542FE7F9E9606A795287746BBC655BF5396CC03256B85699407ED48A03A0CC3B1F46"
02860     "E10D9019BA22B02D3CAE2A766394009A27C0FAFBA48BDEF936014F035F7E81DB4C6CBAAC9F"
02861     "596058D987E08CA8DBB3AB72B8A9AD6943060613C711F1FA7BCF3BB05FFA47095A7D505E54"
02862     "D0D361A08916FF2454195A1E7ADCFE1FF5AD6C6F84",  // PRN 30
02863     "533F653DDE61A6E71AAA977EFF7A55BAEE911662E03DEFEEC8BD919CF1ED7A4794A83E1DE7"
02864     "FAD820DB9D9A9DDEEF8E72A63E2C3279143D9779B3277B35CCD804707DD408CC932B6E29F9"
02865     "AEAD59D7612856E0FBF51DEFD264E3ECBA052DE09D55B5BCC81B566EF68A4C198259CA131A"
02866     "62848D631413D2379911302A3DAAA49DC1CE86DB465F50F5186241C749053546766C757A27"
02867     "8DE9AAADCDFC2AC87373D767CE9F4F6C58147D8C7077A70323D4CC8991F88B7737D894612B"
02868     "FDE4219723CC91F91B67140DDA7908725098D878F273EACEEEA4D8B91E575547AE3FD038A1"
02869     "3CD453CE6970928FA7C7CBBF575EA520307A866C44F9BF74BFDD174BE5AA2C91F1F07A83E0"
02870     "5C7C4474C1A5D87C9A6B74AB385086BDB089466A5945D4C03C2C118FACDB0D1860C506A3F1"
02871     "A33A302922A6CF9F1B52822A669FFF5B61F9E6376CD14C78552F8DF262262DE5DC87ADB212"
02872     "308BB8538E8953916B76D2B9BB2523839BF1B9E2E3A13E4EE516E2D6E487723308D06454C2"
02873     "8B05CC693F31415BAC3511048525A55F252F720DC86F83119113D3DBA5CF61DBB35BC3829A"
02874     "001A989DAF8E3AB9675142F38FC71AD47A366347F3A137ACFCDA8B3DAAA67B3E9B47C76A3D"
02875     "85754FF14BE38C8622EADA513349A483F7888C1387E50464DD333317DA61678777D65CD9DB"
02876     "D2417F588EA20B94EC826680CB6B03C4466BFD4DB7B83EBD095C664BAB158647DE5AF4E34E"
02877     "9E01C8B6D475881ABDDE73997037D4B6D8E6209C5E35D18DCD7D617EED87220487F342CDC7"
02878     "1296E7A014CE7C8A6DC58451355D417CA1B2E516FF94B5E1929F5931FAB6B36B240A890513"
02879     "9E414D311D6160ED8EE30F3B4A106C0E3EC5C3501FF2CB186CAFA8FCBF2B1E1337FCE30821"
02880     "697B6612E5C09A37BEC83F953287ADCA66F09C92CFE7072EAAB26EF5A6FF8D3B81EAF97169"
02881     "A59E7991A0EBF2D8064B9E06F04AB68AC257BB8664BC950ACD5AF05B0AC802D1E1D76F52FF"
02882     "39CE6E6E2338A2458370B95131B9E70AC975B029D072C6A5854D38A9D2EDF4AE65909FC569"
02883     "C4B43442B160F645436952923A7589002169859FCE431562DD5C5AED80C3685B53F2D910B3"
02884     "D1AC44FAAE81467145CAC12C90CA2936CB6E87DDCC8C27B0A8B403A4E68B59E424760CD707"
02885     "E9D64077D31A3E274B26B329C8BFC1B020CABE4F2D1C3F1D10A3F3CDD49B394C3FB6B93701"
02886     "D75F115EAA2835951F689B372C9A4FDA0A36861BE8EC3149E5D9ABDA6FF009C1437F2F34A4"
02887     "C80459C34BAA373EF2C45A86C89E37E490347110F69A51314E5E1EBE98A3EEB4EF098BCD0B"
02888     "77988C55478ACC408A506E390050B2147334B5836DD6B30F3A82E2E83EC39E6F0B12EE5303"
02889     "1F6867C1C14F53E3D53F35777A27BD25697B196E0B93F0CBCBB976FC387F5A312A3A8854DF"
02890     "68228F40B07761E48E179D8A6269194A74FBF5438E7D94F93E12BA1B80773F23950D963024"
02891     "7A206073D5D03C34E8F5891DC3DB058E5637D67E82A7219A29E0F8EB736CA75F53CA566A44"
02892     "71504E78DA8A2F4C8B3619A9341AAD66754D2D55644E5E9A932433D64490B72400C6A6C753"
02893     "C036C48D9721DB697A6BC23D3F6F67C9C05EAA50A89B2E4ADA472B134AD3BB78278FF578A7"
02894     "D1ADE55A439CC314BD347FCAE41F886653B8F1965E18EABFC5C25104433A51677D53AC398B"
02895     "0505317ECB059DC39BEB4A221471B508E309326100B4D9EA303B9404AB33AFE5722ED9728A"
02896     "AC5616E1737CBF7080FA0B66EFA42F9BA01DA14E6D312859F2B465A64BC1E76EA8FAA0306C"
02897     "DC0C29EF13D9AFDD9BC5873B9FE28A2AD938AB2C2C",  // PRN 31
02898     "812B4195D6E1D927AA8F28BB9CCF2FB10405C9689EC40EAE93B600EB88F05A713A19C5B853"
02899     "989A1E3C84334302CD2AF7D33593D9862769E8DA2AD07DE85CCB1504E5C4C6A53B81563A2E"
02900     "89903B28EFCC2A260D2A18151C3CDFAFDE9A83149588F37024D67BB1C52670C18C6F4A3E67"
02901     "E944A62C02F868F7B59C3D1D3966BF97A03F08E9E5641F9FD2654A0B0770748ED2D89498C4"
02902     "EA6F4ACC0DE6F137C4456E98762DC12A9CBE9CC46D5053B64E91650A74D6731AD0C97B9392"
02903     "1D54FC6B9D38E505E06018038FD0F7B6C3AB555C9BEAD41FA1FF8657491762E1DA48CDCE48"
02904     "6FDC1DA420D744C376FB356BCB953E00EE1B14CBA4E39CC8737C6F32ADF53A066A8D9DEE2E"
02905     "130E4407642FAAE0417C0CBEDDEA904EC03DC4A1689B4AD3FB877426D77D5877CCA4E7A19B"
02906     "CB8557F2FBCB536D52B7650599B3A34BB09821EFACF24542DB575986669E517E1354F4A04E"
02907     "72E7E271C2C305C691336C741D1CE1CAFA745041EEB8A604D068CA2A42B97261B5EB81C048"
02908     "FE5B0CA2F9289366D600EB4EBFDFBA1AFF091D51ACD973786B04D02DE66229E78C1EE1E78E"
02909     "CBBFA89D2BECFCE54614CA68DB7EEAE8FF01599B364EA63CCED34E32B23970B0E451A8B23B"
02910     "F5D9074ED47B2E7C7F9E63594CB70478ADBD8E1AC688DAA7F7F8C49AE10091CB5EA9D99A86"
02911     "9F2AB21E3E6F7B72D372B2CADF0D9AAFE8049F39DFADA726A264F63C51A87BF1EFA2B05AC2"
02912     "8339346CCBC81147454391DF6DC6FFFE0C1646E27E8F5BD60F3B84CCBD5837721160FAA612"
02913     "9E528496CE181681FB70E690AB67D84A36D1DB8A059B7C637F742DE45F2C4DAF3499D9A1E5"
02914     "4EC0804639454C87C8C0EECEE734FDD47B834D8DB1F1048904DEDAAE71B571F52152AAB0FF"
02915     "4AD8A2DCEF41E2B6B5ECBEC32F0AA73F79E039949991E5DA837799AE4263C2C26768B9A646"
02916     "DED0E3DD50CFBD9B92D16700FB0558220300568D9966091789879787F5792275FB05D86A48D"
02917     "006F0EDD1FC9F2EEF19781E05EF2CDFF2E266EB84782D4BC345990253054683F05A7C1F265"
02918     "4E835BE154B64D5333B93B906E0DE8820AD154385A3EA770332A026BB1BF4E2483CEA464F2"
02919     "9AA82D9BD32B9EEB9DD47BF476338A47F34B060871DCADDAFE1BBA9ACF87792D0BA7BC0D39"
02920     "4CFE8A871F9A5EEF9471F4DC751708A1E1A48181E988CE8BE87466C482B8A4982F59C82E3C"
02921     "E71DA45F4264B614DA599376C2E451705E88B43CB16507A2A75AFCC0A341FDE0C0A411122C"
02922     "DF740A0A947ED016CFF3DA2815C87750969D869249CC1E017F4E952448FB7809CBE4703586"
02923     "19865A2044730BDA7B1049132CEC2BCA2B74D75A3998847F910846AB186D53B5136D6CD245"
02924     "8EBDA6B0F6EEE58F1477D013F50511E630156491AC2911FFF72349A99642857EA7BB9BBA2A"
```

```
02925     "90793EA3DFB7ADE08C191EFC60143FB6041A08A886898CA82015D2FE1CBB661897D208A043"
02926     "0CC55641CC996BE7F3D6B671BF23E38FB63D799A8582715272D57942EF81E6D5C61D999805"
02927     "D29B0445E28EB628A16EC2A144603A2453604052FCAE95A4C98D996DF79A10862B86341B50"
02928     "7C02268E79A8028F00686E8585A70D68A8ABBAC43A24042457BDDB5BA8752F8635409B91B6"
02929     "E880A103468EC26FDD32B514B419C6DAC5E29E6F40D9D1FF2B3FEF73971B2AFF38E2E5A80B"
02930     "BB543FF5FB2B852773C3226AB4E642D8E16498C5BBD10CB02BA51B6BB2A461E7B57A3DBBBA"
02931     "7589BE27E68A68335651914EF449057ECB16B45BD270292AD49ECABA0F9F2067753FCC0121"
02932     "6E9AA55F177EDB2926EEEEF7DE13FC5B019425C014",  // PRN 32
02933     "2C4DE12CBDB7F53E4926BBD2EAFCC67EA4F95146407343F4CA19ECAA81C5503E5711D96F21"
02934     "DAD8D898E60D5C581F63EA7761FB345CC75511CCADC758C4B52DEB658F205D28D9BB6A76CE"
02935     "2187B3385D843F47D5FAB744ED368FCAB9F4AC5609D786CE6D12842FAD027DB37BDBA39AA8"
02936     "EB0052154B57F8970162D7912C186F287D96C93DA8B262DA07C712EE1FDEEBE9F1428293F7"
02937     "257A407B23957A2130F7C1C603102833E3BFBD5F7C1ADB786508AC1C77A3906A186A0D69AC"
02938     "403F4D0C903E7520A7E60021345DEDCF847074727C8F5C7CB63DE1AA4AD0B5A5CEF88B585F"
02939     "B5719409A13CE94D1961027849CD235B3B13B26ECE180F2AE3BE63A050EE37A580DF114122"
02940     "DB08488A915BD5BB9BF09DA2CB096CA28972DDCEE5E21A4699BEA967D98A49E41FA0CCF185"
02941     "4F4D6FEEA8DBD773B5568FA0D250813E280B4F5B84C734EE10481B9E5B2DB3169223F19CD3"
02942     "8FA990F211BB0F2B44576734778399E10F10A342FFB01FA9D5E1A1F80C66CDE9630A6CA718"
02943     "C1C7750DED2AE1C41DC976AB3BD9978FBBEA6A5B1473374A01E80D8C619A01F3E503BDF09D"
02944     "B1A4CE513F62AAADB43E065D4D99351284AF4AD8FFED53662D0EBACA6A284E5FBBD413AC0D"
02945     "09D5C70EF493191C584C0C9E42A8DAC822FC0B0B26FE54FE3B3D572ACCF978CD29C78AF9A7"
02946     "0BBFCF2F8C5C8DB51CFCD2D4DFA764EF9A5193BC6CF5BDBF6CB0EE950CD484D5F59E04F911"
02947     "0370A22E6E45A1C042EAE5FD0F102F87821AF507D415F06D3F91FEEE09511AB22AA3B4862C"
02948     "76CF1AF16E22DD0A57F64C862436D1031D237D62DA3478ED551D620573DAA33E270B0DE475"
02949     "1FE097F03929E162C8A40507829202459790F9835E2AE3F446C9632A2AA1E8ED2AB7BA584C"
02950     "48392C926DC7663CBC93C3C309D00D74CFBD6C352079132B1A0B7982372D8E516AF5432A2F"
02951     "81BD97E03A761F1A2C03C9564BB40979F71041BFFBCCF8D3CC6BE7128BC4E44B6B74EE6F8D"
02952     "4A184196CD7F6E2AEE80E6FDD43A233C808F5F695DA6AA6B499FE55AE82E392E86F89D12B1"
02953     "78B680B3DFDB5A478BB981E276B439A63E7166B5FDF7E3CE7D2CE629679774D118BCC73E85"
02954     "A9C242E6AF2CF98A36A15ED71BE2CD5A81C0935C1EF36CE67CEE9B9AB464DF3A5568F3FC99"
02955     "E2DB9FBC3C5D71E8FB5673CC92203DF913F26C09B48FD209E5161D28A098432B78C7665994"
02956     "E7DBF7FA305551580CA7767EE023FDD7EC2FAC65E94E0A852FCE2C4EEA0160B8F4A159E840"
02957     "3F245EF9130D1EF343B3DBD8933F58D39B7C5DA3EB73B00D34E8442922C6B84E29179BEBAD"
02958     "FC6EF1E8D1530364140E1CC6F29573196528C80F6FE620A7DD0FA5A8754457D3F6277F76C7"
02959     "39ADC5E505EA369C7164D9AD19BB99B37125835EDA0A74689C5BF9365B9F5F1AF984D11C78"
02960     "3DCE83FCD1EA42332D483929CDC8ED17D69D0085C51292669F62857FBAA3BCE0615FB06168"
02961     "C073607B93FC59F6FCAD00A888440646054447BB2DC5E3899097CA81AFA9C1FB8A2ACABEA23"
02962     "7B94C581230000E95C7D892ECBF0A2D3D61D0DC17293B2A83F9B4B05F28FD3E6973EF2F287"
02963     "168A10E69BF6D50A8566869C7906F17326A88D1629C2B7F069CE2C6CE10C601B380BBEE066"
02964     "022228FBB84556CB34EA5BAB3B0D3CEB1CBE967455B04F03E1AC0A214ECB060E14682A84D3"
02965     "C943021366633D7AF0BD663BF7D82661953ADBAEA04841C4EE8CD95EF29C43A47B1BD33DE7"
02966     "A44AC193313E79D4E2C4143C3DED323FF4754C948D0DD14F75A3408AA227190FC21E3BBD8E"
02967     "E630650FBBCE27CD298D925A4732257B2832FB4948",  // PRN 33
02968     "759E2CD07F623C0952E2CFAA733D9651B4D835D66AB0F25594BA0EBE210F9EE20994E1D1CA"
02969     "A64B5167DADD1442FA8C83A4286530D52F2782938D0631396035C1982E5FE67B5A13F1DA34"
02970     "AA2CF78997B562791BBD8368FB5111E0AC9AF0B11FFA6FAB33AE6E547F3083B138ED810FB9"
02971     "77576187675115 51E0F3917CD864D5FF1F58D33088F8EDD6378265F42B2C645FB31F4C58E7"
02972     "E355C4B4D992F58FB8DEA14B2F3F9D68F0E4BE882706044C0B2FA8AC4D143A7B5520D89E8B"
02973     "851EC915B5C4433BE032A5386A447DC39D6B9E9A8238E9659405CC6DB308273BFC2791AAFE"
02974     "9FB5E7B32321076E3EC70A2EE9F02ACB4B2B144FC625C0E936F4F466140506835DB7412BC1"
02975     "829E760EA120B27F4AF273A8B232963E6C757B3EF039B08E8F3E7C5929DDA32DF9A6EEE0F6"
02976     "15467D390C46680CA4FB4C7A1CF0FE06DB55675F6182A37D2B83139200073AD07C3680C85B"
02977     "EB7C494952B4FF7B6C855D8C59A56DD7C2888A5E5321FBFC4D8A8C37B3923AB5458DFFE0F2"
02978     "7400CEE41C9AC3B9C3767DDD07B46F963B6A2B8B7AB378EEDE6611B3DCFB1F70EF207D1D81"
02979     "023D45765F39597FB8D0DAFB8BDA1C387ECC2EBF1262CC7AED32BB9F96A8AAF5A2DF751C38"
02980     "3A486F3FA629E0E2107875F7050FDA5B0E79401047F05BACD37992876C0A64D44E0CDFCCF0"
02981     "7DDE9BDBC9349FEE9C8B54489FDAB5D4A69ED3C9FBE2E4B2389DF0371E46D330B84FBBA718"
02982     "1F2470B1AC1C4C1D60CDC05584308DC7AD76F5DE8FDFF0A4E80335A3F672C29542BE6B31E2"
02983     "8C4B35826694F43867E186A7F550D4C001BE786FD3206A13AF1D2FB0274DCF848C487E3598"
02984     "FCD26E3061C37DFBEEBF3EF29A40BC9F7FEA2CF01B299E511A640248881600CD04D37F0DAF"
02985     "C0B2F473567FCBBA952591129FD0999A512AC95819F12D05D4306C13EAAAD30309577762D2"
02986     "2F461B0AC5C6EBB80F0577EF8A1C651241B55765D5E588D2B88FD661298C8D0B0E6FA1FD04"
02987     "36844C5E48D33E5F05B6023B60D445E85E75A7679365FC8103D93EF86614DC39AC50434D33"
02988     "01B36AF1DF2C9D0BA5041F7D95F565EC1407213BA3F93D81730D28470D26907CC3AC7D30BC"
02989     "ADD886117768B845500E638BF4B2CCD6268A773FD06AC11BB6FEDE15ED2BBA7465DA43AA83"
02990     "ADCAB61D20CFF0D24B3E795D66F9FD59BA1562EE2944CA89E6F710C6FF9E58E8CEA42326C8"
02991     "FF4CBA582A812AD8836291A95C743606BBBE8625F832D4B9846590BC5BA7FC00CA0BF51509"
02992     "8FBC513F15D74CFB8E794FFFDA3A36F35DD7BBAA4D557A2B27959D54326C3A2FE68977AD66"
02993     "D2B80089EEC8B8D0DCB26253BE3028761F96D4B4B041620592A4FE11D007F2DDE67DF840FA"
02994     "A4D32B4A06A607298089C2DC9BEC00DF1D513D94BE286A9FF1E997F9900C39E8FDD462A4EB"
02995     "AB39D05E61C079C1F7815B4D2985BF0855148398A8916840087D9A5E8B4CB61AFA78AC1B26"
02996     "D99F4DDF43F672703DD471255BEAD3566995BF0F73553284D94F515902AB03BECD088C2744"
02997     "D3E8226F4D92F27666671F6428517796C5ECF0565BDA6119E2E4552C06EC615E8708D821BA"
02998     "EF1F372C3E457553673F930220B6155D5D65B7875979B10CE16CAA10287E88A7BEDD89D2FC"
02999     "EBA87B7FD2C5268FB376987637B69F6DE3D7391C767F2628EB499969204978C9E5F6947F23"
03000     "0201790707384D0821D325D48E873734159B82CF59805F0217D7EB1D40DC10E9C60968349F"
03001     "1B513BF7974248E571EF536CFA32300721F1CEDBDE15137DF780BE77607170E55945067A0B"
03002     "1F1F30DDA7ACA3A06BE60A63D7C67F1B551B830E10",  // PRN 34
03003     "7E643457BE98ED5901D3530503D42D96D9EE39DFFD02D12F0BBE04F68E1C5E8B77E1247BA8"
03004     "4E8BF60C260F4DD306136F61439666BB07A7051046B44F7AF36D23429B1C3F54F782962F71"
03005     "419E71897F7A41255E13C296091770303A19C48201F28379161E2F0073369034A2EB0664BD"
03006     "E67567BA7E4D3BA2C2FA28527D303EB2183870802163C28C61A9BE0C5E01011E079A4572E5"
03007     "C34A3498D0FD04D7558023B18A085FF74E9CEE9EE5E259924C67A507E16AD3164DB057274D"
03008     "EF697E1D1CE51E790FC85AF4BF5B9525E0AA29FE86FCB44390365607998623CE35E8C9151D"
03009     "26AAD60E94E079DDA8C81AF7CE1B97C97C7AA774F56EB725DDFBC26599A50DDC3F474BF804"
03010     "3870C6058263FFBE79F4B130CE511A96A2A6A18C71FB439D126E0E7004E5DB95204756F886"
03011     "13F082D4C1E817E28858DDCB3648AC0192A84A0430E8C384603D08EBC47CC67E041AB7B774"
```

```
03012     "5661767793777FCEE06FAACA8954624CDA4C73E0D85F640699D8765AAAE55470F114FEC7A60"
03013     "1DF55DD053A5769EF190B5EA9FD0C66E3DE2C4B852ED66BCC9A3B4550382C4037CFCFD0168"
03014     "60A1E2409AE75C06D6556627A6FAAEF4096BAA4EF8BBA62404A7E0748ED0EA6F38F0ACEACE"
03015     "612F53ADDF502B1DDB6F71C35D7E2C6B06BB0EB3D9C546ADFD66B47498DC20435E97C94463"
03016     "6F4B0E25585B2C8712FBC610FEA0DD17E0F8C144A468DB9BB71DE49850BFB61225180B95E9"
03017     "9950FFEF2162DE93DE4A11230DAB5ADE80CAEFA00A1F0D3403513AAC273A5BD070572731C3"
03018     "94FAA28A37F6D2ED499D8986B89E63F17A5A1C63515129D588DFCC2FE2D6398510636C1A7F"
03019     "1AFAD0A73CED3FE41117002417003ECF249D9C3AA9D47D1FC2199142D7F7501A57898C5C03"
03020     "F3EECDFCAA4AD93F8C83CBF904080480544CAFB2D7D530FB7B4F019F94BBFD600E19485455"
03021     "C0204C74E2BE7A3F734FA5DAB605FA9A6F17C195445CE4468F4A8BC48D6297FE6D5EB930B2"
03022     "831ECEBBE58B8E388B104F9EB81BE2D6DD87C7DCE9B8C9E865668FFC42C0F50418703010FA"
03023     "08B8706057D4E060EC2069ECC456D28794F4C561BFA5A932D496D1C2A6B4FE4B4B448C033E"
03024     "0D86A653BE48EDC2334CDECC309B8D43BB32CA35A11717E291ED206C4DBB8FBFBB09E539BF"
03025     "A4D966F80671B15098C5D36D3302BFA6CCB265BF1E1D3F9EB07E73529D24257440CC7B0A07"
03026     "C3189084934F068351A82CD08883926347BA0049BD18F7812A02B4E4567071C3E8862EFDF0"
03027     "FA1294A321CD0C6E90612F75340E5F1642BB3F11D0C35BA7789C02A03D73673FB4486DF539"
03028     "B80E263FE9660EE962DABE1DB7505D194BB44E211BAF2D8FD1ACCC5A246BC4524F37CAFCC9"
03029     "84CB197DD377FA46EABB86ACC063F36E386D582A8BCD28CA790EE2FDABD07FA7B1C4CFE491"
03030     "B819302F2D8269BA6806F1D80600936DE375939480844AEF32400923E7D833FB8FFA4B8E951"
03031     "2CA687D00E1659C768445C24F19733F9BC0673033777D430CF6AFE1039D5FD9D968F39BA04"
03032     "2D99C4D31D9B38FD494168F23849C1C3D6612C7CB97371D0495723AA190CC10F237DA87DF7"
03033     "0772AC1C6C7C2C70AF94A3D47B296AB4B3DDFA887FF664676E1B79F3483A731AC0CB65AA27"
03034     "1B9B7A7F5165FB0F2C3F276A0FC9E5AFDED12D5BF7A51F4FD85FD7A997AEDCD0384FD9F3A5"
03035     "50548DC1CFF796171FEAA5AC8C2CCE35DCCFB0D994E035652C8789B8352FEC79AEFBBABBD1"
03036     "1A7762FD01748AF96F4A83E5716971E119C3014D747FBD91352956FDB055C7E5B6483EFAF0"
03037     "BD69494327478D8ED9FCA239EA825A3FFAD118AD6C",  // PRN 35
03038     "B43640BBC9C72AD37CCBBFAB07D9A72216905C960FABBDC71DC692B3E98A32B27C7FD7350C"
03039     "12A537B30427EC4C344FD54E6223EB77A11D6784A7AEEF66E24CC4726C5F11BC163D4C44AB"
03040     "496DABC3BDC325F745B35D27F570DF8EC62B141CC4EAD5A50AAA749088E8C7F0F78B2E32E5"
03041     "9807600BC01BF7322E6A3C81127490704617B67EB84A5D802DCD74DD8A30F6DE89AF245276"
03042     "178C04147075FB682515B9CB2CDE9086AFE2A54B9165413357373FE1EF0EEE00D5AF1D795C"
03043     "394902C4E255697BF14E7B04C1EFC1312CCD0222BC924598DDB55462750EFD69F4EA5B3ED9"
03044     "D6B08993E58AE16798E1C008FA8DF3B65B81962E09BB2CAFFDDA2297753A9D38E97FFD5554"
03045     "CE3B2D61DB9EAEC49555310AFD99CD87C936DE30E06337B8B2A302B786DEEF1C4C5703FB87"
03046     "9711E4556032D6C6C0D9528D714290C910C601AD65D210C5B815E4C1A2B710EB8EEC49AA3F"
03047     "0F311B56D13D7C660106A7AACEBA72F6F36780A4C806E7860393601A1742309EC34C86917F"
03048     "A7AE1BF4704B07046C7D77C0D78796D5C35D378D15175F23B9CE6D5432958D82701A678BCB"
03049     "E19C90334CC8A4E89199094A56A502118AAF9C070254E6A9D0CB2EB85C0D1955FB3D2C09A6"
03050     "63F88B11DDE4DA7360C3730B8DAD5C27C70BA8ACC1266C9E499018D4E860E4A40A8B430F2D"
03051     "7165385D365D3086BF443D63953F269CC81CAD0E0A7C5A49C856B24C0A278840CA9ED5E4F7"
03052     "667B61E42CBD157C9905DE937D6BBF841F36A69332CD4BAD90BC2E5992D4BEA4732D4076B2"
03053     "4B9437BAB310767734F9449577732A3DC3E81185487B57D42AE400E5F551AD00E10A15ADA1"
03054     "6C2634501CEA05BCB63ECD956081609485F2AB31E1F8CE022BF8AD3861ADC7A032DF769A1D"
03055     "E3C86B775C3764A9554CEF675A7AFB72918DA2A4CBA8BCF94BDCA97819FCBD2B89D0F4A960"
03056     "51058852271C5EFBD62ADA7EEDE920F9F333578D1AC6D49716CE28936FED5D9D8D5B92E30C"
03057     "DA5669EE098732591633983DE7BF41797C739A7E8F4D0FA30A60105D32FF913ECCF3F3E4A8"
03058     "2EE93CDA04D4E826C045ABF89CDF67CC0BF70D32CC08656C2D8FD2E23AE40FC2BBFC6BF258"
03059     "F006C1597AC3143BB9637B1B7B0B0ABC513D7F96F13783F44C0112C0F2C35C09A46D27AE47"
03060     "F0ECB22EBAA5DA5C4FCBB0923E461EE0A524DF59E83A791FE9E7C9E114235ADF9F17181583"
03061     "888051571C82A778883DF35956A7F3BC550E8D06BB7F158A82A708C0382B54CE19D14794CB"
03062     "7CAFCC6F67FE61C3F6C1961E622246F5EB285667E32A940784042C7E8A3E3D661BA42CFC0C"
03063     "B2414B37F5B17CAB201C1A7F894F07CB02A9EA757342F76FED55DADD334EE8BBA9C9AB962D"
03064     "1F46FB47EF3B1AEEFD33961E887692B6E8C209DB819D4DAD1721A061F4A5C1EE753B518CB5"
03065     "B56BC79BC426A6B82CF2C891A999566AC9B3B55D7694112E8DC14673ABFE7A6D5DFC5DF1FF"
03066     "B3C1A44A181D6A087D4EEB0A821C250D0EA92D760A9ECD4FCE4B6501735D33B60F462FB971"
03067     "CEE90C706B6106A022F00FFF3ED9E48450790EC73469089792760D4D7C84A218E11A557E30"
03068     "D036B8AC2AEA3BAA021D0E559084E4DB29EAE90A27EB3FB266BDDC99B3A9918A02C7F7573C"
03069     "1638902DDDCAF7C72A33AA9F7BB6BE755240314D1459F3069746B824EEFE49B2019298FD52"
03070     "41C75EC5E4A78893D7D2B9FF9CA8D0D23D25159277C240FDBFE93D90005E40B2B1B9A28662"
03071     "F81FE38A04481704213800360963 9C347DC84127EB07966451DA55ECB8EC6AE03F23ECB42F"
03072     "02E19FA2CA20DDB145850D2DE8F0F8F4334CFD2968",  // PRN 36
03073     "05671B4FD9D018B1D63A7BEEEBA93AFE4053484D26C376C757B1DDF83B3CAE766368B02886"
03074     "DA4BF41DAD163D5610310233A45694AFDB4C7607F8F20FB2D4791F20E73206A8AE3DF636B3"
03075     "F0BC8FF71DF1A8508CF7E369F3949D58EE11E187B6898FD239C01BEDF4EFA0915CFE0B1188"
03076     "49407D794B8EA35D01AB0003E755900BA5B1A1207705AC2083E861E6A9F7D9D9E688A34291"
03077     "20B95CE62698733522093BC64EA57CA48F05161C0FCB369B57F16C449D3F094C0351F8717E"
03078     "FEE5A576454D3023DA435C8C553F4F786C505A16B9DABFF43DE50091F27D338C78AA5ACE5F"
03079     "219F0A9E0451D0C5274FCC435C5CFEDC6641D1CCFA50F2841832895FE01A95868661A8808A"
03080     "B763FE2F8781FB92EBF1E946FA7ACEEAE032B8D5AA70933A05B361E275DA3C0F29440A456B"
03081     "2999B11450D519B0E3BCC5E609F20F04CD1C65F35A98B69715A2599EACEAFF382FBD6E7F92"
03082     "D3E3BA84563551259A186915B9EDF350EBBD785245BE26ADCF2556204827028FCC37304848"
03083     "C04BD24709492F5DB034BCBB7E56CEC0DEA45AE9A4252D0510C1AC90F31B3B438D092C0444"
03084     "710467B631853766ACBE05BC40776A149F6CFCBBBA920F7EBA4318D0BE53CED07E33C4105A"
03085     "C204909FDED308608670A735FE5A6B6EBE4435A18A031D4EA332912C16AB7000E353ADB0A2"
03086     "659F09846A552B93D6DAC1A906DA744518305C2280B37691BA418FAA976F301D49E091B5B4"
03087     "581FD89072FE7E670048AA5654A98AC42714E55D5323CDF0417F51ED4560AF43C7B8C5047B"
03088     "EFA37685D0E8B928490121344AF91A907BD17932698CEB7BA24EC0B559E151E4EE69690038"
03089     "4DFF1479BA348CE1328292189DB333AA9D6A62A44250ADEDAAE4C7FA208A6E2C53C6378BE2"
03090     "9D294043A97AC937B68E13F0F093F543D3BB039107B4748366EC97ECA5DF116B706309C98E"
03091     "54F88411C7E13653FFB21429559DFD1A45A85189D11FC52A594C2DB73864BDA9326170363D"
03092     "AE363A5BF7EE1E91E87C6E5F5246EA4E6835D9BF6A943082C3CC6130685E7DA91A3B401E0A"
03093     "DB307FDA70B37988E2928B07E5FC572440B6972B430C13053F57F2483CE7060076FF2847DA"
03094     "B897C247F64D00422867A78C29A17F1AD28D2E4EA5FA14596714FF3CC855EEA97E2FA33B46"
03095     "E0E34A38BE1222F03964F1EF6CDDFFAF7A18F752894857F0CA43CF34C21C2E69135F5FC960"
03096     "BDBB8AD3481209D9FFC08CBF3C2807CF6BEE56B4B91D3B1980BEC5F52D6C56B4FD36EBCB3A"
03097     "A02054B1BC1EDE9A3DC61E72F1D79874F0808FFA7A8F2FE1F183B77F4CEE38C9AADC2995EE"
03098     "6F3613326410DA4BB99F7B33B77EC19F65C0072C33A38EE623FDE3C9A5D2D4E7C539779A15"
```

```
03099     "1C1D84B109EC2519EB625EC61D2ABBB54C3C9F7629287DD1B2FEADDD16983FB9492D4A3322"
03100     "6BE55804A4C7277299CD3003CFE8F2E7FA946F749DFFCCE465C7B2BD251DC4B0779D543CB6"
03101     "873B6DD03F82B09BE43D8F114E2E71F10DED5AEF9EB1C266C5A9B7F21CE1DE0FC6A86FB705"
03102     "7FA75FEB098E041C95B2D1912849BDE26BAACCBFC105BDB8E6BCA0B8F0B88255AC0DB9854B"
03103     "FD8526E51656DE55E7968C4FBCBC1FC32D8BCE5BE8F1B132F1B24FE049C2B3C18A8A7C1375"
03104     "9F725AD2957D2E73ED0BE3B34B0B8A14F173380BBB9CECC925086B2734982FB234F3940E42A"
03105     "82FD196AA218C1D488915144D8653B5EABFB18AD35223E35CA403A17E8FA0C1028CD048672"
03106     "7447280E2CF9046EB60607A6AF84C7ACC03A7321AA3F3787423B24235A2191E0F13665A3DC"
03107     "F1F5C0E964B7FCBEBA511246AE50B9DC1EF7254C30",  // PRN 37
03108     "8C6FE0D3BF7763C15CA86E3AC55695028D02B6C3411F6DB117FD982A5B7F97D0C1C05067DA"
03109     "46BE7FE39687CE4DCB4D15D89D3665A24932CDD7DB871481674DA85C2982F5C896E7F68345"
03110     "DFBC22CB465F43503EC3D7B5E0A4F17169D7D94D3A64B27911DC34DCCD3AA7B3CE2D8C9A05"
03111     "D75BFCFE4BE5CCA3DD71E7FD059FA910B71A2400039A7EF536FEC7E93D4C386E7ACF1C8876"
03112     "A27A2B14DD49A0C6CE150BFF180283B6C3CB228DA9AFCDDE6A9C17FDC03FD7D5D28D8FAADB"
03113     "B2C49CDF63D269109EB0A7DE261123FB15D6108939B231070F60E174D30169D2155DC77966"
03114     "1C570B5092781AB64B122C14CAB4A511A493DAE03D2A8ACB069F1FFA0E83C03837E62C8923"
03115     "159E6140E237885BCC6675F906CBBEA9A5B5A7B6AAFCB7D03C503032673BCE6016096E0C1A"
03116     "2028D99D61FC40EF0AAA3ED0911506214B0BC37F6EFED7BFC05AFA6C3412FE72DB52F82DBB"
03117     "0698B9AB4E5219D072CF803D52058E478324C523D765E70A277959F94692227F3123B9DC1B"
03118     "EEDF124518C2362D35229ED62CB10E32CA78B19AB39B6B45D8E6B8AC9167157A715093645D"
03119     "B3B545484FE7BC8FD96CE19DBF8DE9F9F7B8F49EBF257472116CC6E453D417E0DBCDD11D2F"
03120     "F7ABA5851DE6CBF879B3F61DAFC00EC007F9BC702A557B200FD0D0F3FC5C86D824B2BF5692"
03121     "117CFAEC14A34EE434AD48B1711ED9664BF1C5DA02595FFEB35A1963D41365C713AE701AE2"
03122     "6B0E544B19A7D6495A9D21502C43EE1E7A981F2EBB1CEBFBF6D282BBD30B545385F58797A0"
03123     "ADA4435940780881518EA6E2FE6B5DF96ECB4244D183D53E038B09FE6398EE7CB4D04FE340"
03124     "C370074190316EE51E109740EB87187EE128F6BD1B8B263D0DF31A95B2FB51CBC9CF34D75F"
03125     "70132970 69EBF439E4E5F7E8DDAC4FE0D6E4B7BEF0DE04F6759908BDD0B787523648AE1046"
03126     "B27D06DD342474FE342EF6ECE8B156CDD8E72BCF8EDF8D4C9D1E8F84B282CFECB2BC08EA0F"
03127     "86EB84D460B4F2C82B3DCA5603A76C46C6D09F213B62F8C97A150328ADAEE4565ACAB5CF0F"
03128     "F721DA5F86181BE382E860647E1ADAE0A020907DB64E3AAC65E916B098D78524C9B9EB2F20"
03129     "2C34E984C44C8361553AB85ADF025B5DD735D175B7A9D202844AAB1990DB99F93D92675FB2"
03130     "21E99E3F1F202CF72DA0D0E8D525850628E03FCFEF6C8C5DC04E935FF0DBFEEF1C33A75340"
03131     "E07AC5F4374195790B1E8E044971051F714988BE444407D944FCEB9E6C0E0C406E8104290F"
03132     "185B0D65B3F9C0F1F8C682DA4325FF0D9181049F2BB7F2235FA92673D6550FA1CB222390F4"
03133     "286116F6BA9603F8BCBCEEA18BF472920079F0D2AD480A36D1951E9F1D25A9D9F33D3821C2"
03134     "FD73433CB6EF1509E358531E14DE20448FBB68DA0AF81C95CAE30E4B65A95B5542533072BF"
03135     "B07180CCC1E2783EF541CDB23C0B6AE07ABED0E4816A8FC50911B41B56A8C29A20C26BF5B0"
03136     "08164524DF90D50B4D5EE3308FC42CF99558EEF1FD32EE9FAF0F1D1CD0CB40876BAA24C70B"
03137     "8F74D7E4B3A963857DBC6373C3681767C87E8EED8BF91402D157619B7A22D1ADE73002210F"
03138     "19303781021EDBB8678DA736FCA5253174E271BF93F55C314A44B63F19475DA09C4F5A5577"
03139     "663660FE55C54DE04DAA4291C9545554F8C59299560BB9FA3EFC8FCF2FDDC328526AD3C0E9"
03140     "E583BA70F5C39CDBD23F39E4B7D190CB07D74CFBC8263BF952EA8666F98110375FD8B44344"
03141     "BE1D1ACE3E64D7BC42ABA097A8411E6BA288489141957BF304B23B9685500CBF0728F1E7BC"
03142     "2622F8DC693A8A6A188A57838050974098D951053C",  // PRN 38
03143     "978D4E016D006EBBED30E25C1E0C5843FFD41A9B72D83E97C511FFF9C43E6EE09C7D3EFD6E"
03144     "F5E1A76184EDDAD3115D2C7679B2E02F01F6B1E96E2386BA2FE579879C428C3ABBE9D8C91C"
03145     "0C71DE099396744BFAB7AF2C21F4DE6148B34967F9D51327B23E207C07F47BED4B15327179"
03146     "44D203C29BADD646321834B49B9B20F8C5897B20C2F383202ED43A4842F76BEDD254B072FD"
03147     "093E9ADEEC149E08892127E97F9CA9A3AD97C59B8446A880D6766C2FF2C1D596E4AF1ABB47"
03148     "E5F70EF36539AFF10FF64A2A5AABF499DBCC1DA9733D3D374A17FB40218EDF9048930C69A0"
03149     "33D3C4AA95F7B43E6272A138E501FE02FF900DBB21F5BEA5219BAFC0B929D1757D33DBAEE6"
03150     "341A78651801931C18C36CA71F5BB2B0BE5F4F481D49AEF2047DF7CBBFA95052C9139CAD30"
03151     "E45B631F7C1263C8ACA26870AC1C965173ADE2775331146A8ACD3C28DF23D8AFB30292CEE2"
03152     "90A63D9B3535DD0170218A4469FF12A2848235360CCF2F1CA7AC67091178EB993701A8AF41"
03153     "E5CDC5E1857D493C98E144188CCE36C19450B57FF60CF87E13EEB4F7BBEED3946421C7EC39"
03154     "666CFE0B4017ECA70B5659B39979DF346F91E8322EC121EA7A41095BA7234BC65C5E96F79A"
03155     "30104C74ADC4B626C6B88021C238174087B86C1EEA1ADFFF81CCEF4E9D4205D20772524A21"
03156     "88F7DD8C4F509CE8DA71EE773237F39C2895A2378DC6433E8F5B4C9F54F7CFDE18C8C02FD1"
03157     "F6A66519A35E28016EE448178B437D32FFD3044FB513B913B2172F1B20174D344EA0B2C0D6"
03158     "CCF0B71C286A530126083AA283BA66C5A18B37D3C6DAD86A2E95ABECF1CF0B31656F11EA5C"
03159     "97A29883C9DCB3C739353FCBCB2558518D282FE6D18BEFEA9F38576A912376A3D4108319A5"
03160     "F43C5A78DFC2C609F32FD639BBA98CFC4BE1BAB7E4CCFA0C75D4BDE947DB3DF9753B7FFDA5"
03161     "8084C685DEB8EDD82F1503F43A0F8C25A232588A8DA22810307FED26056AEFC2D6795D0987"
03162     "0D3365EFD81974E19D6F1BC0033382D85175F218B35CD607410092F52AF3B635F8FEFD32A1"
03163     "0CAD30A036ECB9C08456D9A5D676B15605364A539F3E3BD881D86512D0BEC541D2F679A65C"
03164     "B0392101681CF68A63DE7EEE31648F832DCDDFBB74FF3848C2910997AA3E8ED3954C38DDCB"
03165     "2403DC3FE8E760D9903E4F0B9479CF2C5EA7B191E1B2C353E180FD35413F53340C89A96A44"
03166     "FABE322C73005A1FA423D2EB197089B83BADEB70DDE9FA1354B964C7236B4270DC31523340"
03167     "0972AC792178FA4190CC132E9AE55FA4768FE302E0AF7220BEA4C82CD04ECC17A8675CCCEC"
03168     "BF2CB4EC6EED8377795B391DDC3FA3C59153B0C3D196369D7A29EC2EBBB911E75991C56872"
03169     "657C853A6CBD1DC429CC745711B00082245D617C2C323A86B73A735BCF2B3A9ADCBF76B991"
03170     "6680E1F35A59CCF6493F58FE9950059AC4A26B3B26282D8AA24A0DBE150DCC8A7A455DC82D"
03171     "76F8EA2793A160FAA04592B2D85D4095E1E62495DF2C19F21D61D2D743ADB2833B199C549D"
03172     "782775549992915FB4DE07978E3A08E441FD938008033295CE4840AA95282B36C02C786F16"
03173     "F51D935108355D791EFC2AE55270355DA526CC1098FF41E179BDF5C72E5B5BCF38689732DF"
03174     "1AD18E90C737D2DA573C0872688110470938 05B01FD8082741E2A63B4C6F7B38644000FD10"
03175     "70A470D54663C5398B2AA7BD0735BEB0CAEDDD8B7320C771F06CFAAF8B69ACB189EF86D9AD"
03176     "344E9EA4BE14933BDE687D9B13E29894CB2720B28EFC50F9E12173F8072ECB4E99960A1777"
03177     "6459ECBD324AF8BB76C9B8FADF41C2C0398030FCF4",  // PRN 39
03178     "5319BF9795D7FBB2BB137F880EE3295AC2E4582E7330721444D4CEF185D010FE362980CF49"
03179     "AC68918AC66E8DC9DDECD857FA69DE0989E1C1BAA7FC525D07D84E07329FDA7A0FEF713E9E"
03180     "B3279139A286FFD12AEF399105192599C5DD57B18CD6EAE028CF605B2DD8C774954D7213A0"
03181     "109C34B23027720514B6F96E83E4A2798318AEE1B3C42296421572AD26784941110DDE0F02"
03182     "AE847BEAD08EA15D271CF8A6F21DBA4D586F9E2A6729684CD27462AC84C3A8B018DFFB46AC"
03183     "ED0316A27B3EC714933A131158746A09BCBAF89F41247460990CD87F4DF22776497306C644"
03184     "34F69462CEC23597FE9FE0B270D631F0BE33C6F11CCDBA3C45B1F71BB159ABE23F15F6C3E1"
03185     "ACFAB35B42A839A7ABC9CB05341B206CBDE37BAA1EA7272BE207F46707139838687B69BDA4"
```

```
03186     "76E44971F1E893804E04FC8EBED3F8B86294CF330A34066BF1AD3AE20173DEE10F94C8A11B"
03187     "8E6706DBDBC86376A400F1DC2C4E5B33043C643887FF010443444F6159D87C6EE703C505E5"
03188     "5EA810BB9EE3C65381768A1A4B9AD208A4896AD730990EA950381F6D93F53EBE814ACCF92C"
03189     "D92DFF78663D43BB4F089EBA44BCB30522237EBE9EC5869E8EFC22326B62BAB02D863A7976"
03190     "0AFADF69BC66F1CBB164E7763FBFB2519F8D99D0E759BCA234E4599FEC0985D9A053D9D733"
03191     "CBBDDEA4952C08A33E9180331F557033712284022AB57598788C8CE922ED788C4AFE1D3BA0"
03192     "218EDC4CE5ACD412A6CB1A078CFD47538A3FF0B0AF195B42E1A51369D061BF4C3FAEC1B383"
03193     "43D5D9C4ADBA22C906DE1D1C419868BE94F2DD34BC5AD40F14C288F2850264DF35ED3D9096"
03194     "EC88E9065C5E031541BC72C32FB151840E687B697745EAEFD00020567F7BB2A3CB18EC6B5D"
03195     "C4E247CBC5E93CE93795BB296D0A50F835CE840A0672A486C95DEFE9CC3DCC1C2866986828"
03196     "B46107708BDF0795ABD57DC40CDD1C63CC66F9B0C58959160520DD6B345CBEF24D9796E102"
03197     "59206B932F7B42B8EBD890C9FFEE5F7E149606B2D6760D7C94F0772BD7C9A32785A420FF8B"
03198     "11854022E2D5082C6FA1ACDDCC173CF98A5E91DF8D82F0BBE27D9341F1AE32FABF650FD800"
03199     "2B0EB123ED813216D464E3FBB0355015FB86CC9C5695A8B4DEDD117CC6215FE72DB4C1D444"
03200     "5C5408E6D004B2A95F5F222D1428931A556C3DFF7CC46BD0870ED8D8396703DA8D16637585"
03201     "425E0151523E3122E7C4ED986DE71D41EB795927C0E7503BCB36887E6925DFFE37C9A37CE3"
03202     "14E76CF8902E63AE5C652D7BB7785D6B64474FF02D28535360CD1CA75596FBF3ED736F2AA7"
03203     "286950DADF5417A03509A53E883C21E1ADB5AE2BB44B2AAA8164976FA5C258BF0EBD5F2B06"
03204     "768ABAB6994AC24273DFACF41B6B8314AF2F7E07E6CAB02AB230B854016BCCAD1666E4A403"
03205     "641A8C1BCF818733B1E08BDBAAB18C165DF851BD18D03C4C0E6E1FC48DCBDD1E6B5AAA263F"
03206     "C5B7BEFD4A4F5CD5693F3C223B4963FFF2BAE559588ECAF1F42A10ED87A229BCED177C0B4D"
03207     "BC197A82131008D351A580DA5001615421B12E1113D7CAB59C0D38349663D898ED7F6F2913"
03208     "CE4275C3C4BAB98678D7757FE101715999389 1836AE3DB266F685FD9E726374407C2B86F4B"
03209     "2AD6C8E9DDD5FFA858CB9A416605A45EA8A0F5F7794925A54B42C60BD043AB0C5F18247400"
03210     "414CB83401FAD574425F8D1A5D5D85B5D768915209F2762F2FAF53C4A1640899C0BF97D952"
03211     "B52ED5DB34D3F9CFC9BD5425071F6B564B913790600E6AB98C7C100F4C460FCAE4583944CA"
03212     "DC1CA5B7D85D876C88AF100598722CCD82A016DE1C",  // PRN 40
03213     "5164996BE5360659B3E767D910E441BF5E28011571EFEC8423A22BC7B7C959A9E532A3F025"
03214     "D090778B2664222DDAC9889FA15C5FA0150C8030322B9DAF467B854D0FA7DEE36BC529E596"
03215     "C35D067B84AB92D6B20FF2AF5FB08F2276D7F013A796783C2B7354B92E364EAD733771BF18"
03216     "AA545E1358D262918609D3000E1E523161BE3A1FA10B4C7F532C5FE97F7D5B406A6DD8C5DC"
03217     "FE82355646D4CEA439DDC3E61CE0043A8C4EE5B2229821DD52CA33A20FF386A8FC37D32EE4"
03218     "4D94E17279536DB0285FC64B43EAE566A52D369D867090A9A9D507D78B67D1A10AA5AD5C6C"
03219     "2F1BBE87A5C9208241DEE1C156FC9EECFA6D01F8C108D60D08036197015D0C96CBA8DDAA2D"
03220     "ED0BC52A9A04B74B9123581825B96781CA3F1CFDE320823AFA3201A90A50FB65DD53DF6CD5"
03221     "3D6442D6D0630E920D4EFFA664AF6CF38674A177275685C4A81A1CD7A600E847F84308D76D"
03222     "94D05489ED157707732848896ABCA720CC13362880413B6536A8BB9E43D29E3FB4B1F97715"
03223     "285A68CA9090B5060E53E4C2CE4AA3C0845F4D69F2BB00D26541F58AE97532DBBB1DEFE758"
03224     "14421128E41139713422EF3FD13E03D75AAA33B67A518F4B0DBC76BDCDACBFB562BE4D4598"
03225     "523EDF76B409833381BAFAC5259448B15B37F4C031FA60A64ACB67220EB7DB531CDA8BD7B8"
03226     "2D4C3BF9C8CB9D2E0A7441C64A329A7EF5E2FA321C67E5DE16D82DD36AC6098A7FA12D7691"
03227     "17DFECD70B673C97F2D8147B7B9F710AFC887BB953C486E4A446E7C3B54B6437EA5E246D7E"
03228     "79C0CBB9F767EFE31BF66D9E95A1D9527F05A84CA63B59A09BAA537E97BEAFBC5648CBC3B7"
03229     "FC7F526BCF287D49DA8BE2E1AA2C9518B9D2E67CF6143579064B33715D11225FD754F48031"
03230     "DB3373733D1035664FC26FF055E98288FAAFE92C8623F8B0C2966387514B649F4EE8BD8550"
03231     "EC31E9F11415185413354EB92DE3775E8F81C6C3B11ECDEFD5F46F3C295214DAD2187A5848"
03232     "17BDE578D78A0D1BF64D50484EB8C691D83F38E7FBEB0FB38638DBD91A4E82545BCDF4740E"
03233     "C9B2A467F6E6EE67F8E448EB7DAEA82EC60C0DFCD981729FDE9B8DA994732ECD442874991E"
03234     "5047E974C11B7162C792DC7745C28F93A195143F5C4E2B2255AF0E070CD9D54B28BE0ADB8A"
03235     "1CA23F2C0996AF3B5DC54FC581F0D8DFADB3EEDA4F25E25C1E54E25E079F121C2132710F1D"
03236     "844F95404DC9DF30548BB942B409216E2203F7B723F37351D86A49B2EE870CFDDE82614627"
03237     "1B7C532E9ECE3C7E985B22AD3DFAC4E7813CC8BDF3133AE30081A1364E4203D3BF3617B4A4"
03238     "639AFEF78C4351A50E5643AC44B3C51ECB9EF1B1073A6EAEBD7ACAFC57DEC2781868F535CF"
03239     "0BE6318933659FD643D8EA8D0189AE66A0D8EC34E84E043313A8481367FAE12F1CC684949F"
03240     "48CAB82097A50E238008433AED289C5B948D7B1A42B8074913EB531DF7196C40B499034D15"
03241     "C81396B41A5E5C2AE85F775DB18E090BE56FCCE74C733933749C07CF695013FC804A1FFC06"
03242     "D4DDFEFFDD235B6EE9E0C9D0BA161E83DFD786EE54E424B8272D1E167F53F9043E2E3B5934"
03243     "600C0F063BA7717FF53ACB74149DC1211FF1AEDE82F261ABF6B1C58259416F3DB7DED56F11"
03244     "D1C148BF43F828DDD892805B080C3C68BC6316EB1C0F533B9ED6C81C48DF418FCE514DB7C5"
03245     "8783B5F4C27AE0C3BD31E069A22CE0A95AE1C49099C839C64575D897A6AE55691B71B406B7"
03246     "C6EFB460CDAE1126304B5A3D1BB438A9C2B7508A28DD58C42EE2A6B290E88FA5A8B0D77F27"
03247     "0EBAB52B0CCEC8C0110D69F8314BBBFBC31CD5FB50",  // PRN 41
03248     "6E4292DC48E550CFD81C779B5DA6D2641C2F34C9E17F5D2D3B2DAC11035B1EC0F02898869D"
03249     "DFAE322AE3AD6917A37610C5187ABCD89B8393081 82E7FAB6AE980157993F5371787BDFCCF"
03250     "18D6D6DC9376FD44C330153556B3D39F9BED9920670745327918B0A10309F40E7668C590EF"
03251     "C0242B002D4D33E4EE9FEA4625B98BF9873BB8D352CE0ED298B0BD2CC251F097F722A28383"
03252     "3ECA53921476FD2EDFAF8666C198D808A4935B3DB5C2CD84918B97AB5E1200A052D495B0B5"
03253     "BB5CAACBA0317575D8C6390BB21F3463893C931870DBF2C2C9AFECB410815C65EA33AA0A99"
03254     "6D3217DD1B2DD826715DC57E103FF6166ECE2A3BF309552E755E7E41F7FE28D1CE1022913F"
03255     "BECD9EC9C45092131F54085B987D03750B35C8FA42128C3DEEF620AEB3A881CC8FAB17F1B3"
03256     "19BFEFDCDD3F65A0CA9E79A90857F6D3DBB5EE50580519645CC41E6CD359EA97E2E4D6DD90"
03257     "CDEC944DAF3B1C461172BC1ADD53451574686FFC8FA6250EF3A82736F120F946D5B0721244"
03258     "1087AA64BA01B99CB5648CA33040A6DAC30848AD2C3DD96160626018D5FF81D6DEA41819C6"
03259     "FFC6C1DA8CEC6FC9C35FD109D571B800BBC4F43BC552BF5B05867B65C4582444A139BAE9BB"
03260     "825F9B0EA9AA1540B0DE16CE5BEE092C80A32C0854DF3FEA26479A93FD5443549CF03447A4"
03261     "B80F1A951ECD3B5DBF726B91150D53787ACFD2C6FC75804EBFBB41F2AB2C5482E7149968F2"
03262     "94F315BDADCDD25BD8469CD679D401040C5BF3B6E9E3E17D4C4855215E6ACEB2498A663F34"
03263     "8569718E1CA6CC6981A595D5A81A7D169BB0B928B2735C4D1BB17807D7A7954E0F217427CC"
03264     "F89251D039D1A2195F6583F76F0AD31A4096DC5036AC56EAA6EA9253DF4EFD1FDEE25E12DC"
03265     "4C661B1D472D890010FA64ED067AA64B9DD14A36D1F8A5BF98380B273EFFCF0AE2C4E9D3AD"
03266     "E8342569BEA96945682969CE3853B26845DBF403A3831E3D2159C80D2E97CD1DB0AA41E8A3"
03267     "BDF1913E5E95B8C487D00FBE502C82F10FE800D2F666796724E39E1C344EACC8DDE3803D03"
03268     "E1BE070E04B7C1F4795D585EF7A015481A94DB61155BE940819A5471FEAE67859F5C255E42"
03269     "F2A2BC6F0AEB3E427ADA9C69BD2B0A441246E85777C76E6C9D5EC5C6D710348F1173C78DB6"
03270     "89195D3B48518FB8055BB4CD09C9340A4A1873304F1476B542D89B8D3220AB5E29839A26F1"
03271     "2D898316470A96D271BF9AD0E42E77632351C92C1BE7666D39B6BA0B8D970C81FF3FAAA6E4"
03272     "630A9F463125BF6C71EF61F7B28AD51F3DC63F370A69101125C9A319D58596A6C8A8418077"
```

```
03273     "F49D2BA213EA67B60FAE83770EF2DCA7991A1891AAFE767BC1BBFE388E9A0017A34757209F"
03274     "47DFC254BBBCFA0C9DF4CD7E76FF37712F0A45A04323EB64D03F33E78FAEA8DA77B1B03CEB"
03275     "E908B3707162BD8C5DDDA3536B1E226637E34B50B1958AC784321F99B4C781F6F940FA3F79"
03276     "16531CA26D4C2EF4022B2158F6F05252C923506EE630C94244FFAA7F1614C11F63AE39C1E7"
03277     "8B8E9A1C526A98536C5453D1FF66C6ACF1E63C9B8E4629221FBAABF71E1043024106445FD7"
03278     "E0C1FB82C20B847069C706A64ACE26624D38F2AAB85AC3D5C90320AB63375492541BEBE367"
03279     "86C34E7A199F97EF7C6BD4ED7C0700239ED0DDDD008A9C423984E60B2800D09B9E9ED49998"
03280     "90136F322429BDF2F7416D39AC24705AC29A6507EDB8E62B7098DCEE4B8B3435A918B9D6F8"
03281     "1E1187B712D0084E7F29D61675BA62840ACABFA1CAF7E8D005CE998B9330508ECEF8A1372D"
03282     "670D177C6DE1DF236E41FD37B76D899EABF6CD00B0",  // PRN 42
03283     "DC86A357A9D6AE4E3BF7D5910585996FF179F01F218A8BEA0CE0B798C277106D1AACCBD428"
03284     "A267C36282AB20692EF0C130313CEB5EFF80954EE294400687AFC89B72E20B413320314B09"
03285     "A47E9C6D8F985DC3B8B8DCDCD4A67B90D989852E4E8F073D0AE845F3A6F86534BA2172CAFB"
03286     "DA9CFD65A2EC3FFA376EDE069E3B15B7799C8192BB720BA47202B88CF71E4C820D90DB020B"
03287     "BA968510D0C31409B070420DB16E6BA0BB5D67EA3FC3F4D5F46F59599C11166AF0A52E35E9"
03288     "EE7EB3FED65FA5B624D2333F3CB13FEA3431207E7A9CD9E6B0BA8AAEDBF2B398D7E4A3AB7D"
03289     "BCC8BAACF5D7A44A79C2B2D1D2E2790D10C84A1BE37D4E9E66B749A270F1C78989B0E019CF"
03290     "858AB22FA6994A05E3A08B602F54B2E5C17F76629B0B775E093110A1B3FDFEECB68191B36A"
03291     "B5B2A33BB733310E74E1EBA4160DA73A1B9125710AC3C127DF10A47DE23282AB25E28F2F45"
03292     "80B949D5FA060D1C15E45C073CF4550A0026A0CF5FBB40CB25EB723DB6887DE2404921662B"
03293     "C705ED718EA6490ACE13C42AF92A8412EB9843762C0B8F93E05015FC672CFE1E28ADC21EB1"
03294     "12357F7B8730ADE449135D0C27F9CC593AB4331C8411FD1E0139C5E5D73571617A17103967"
03295     "7624E3C217DA037A17F85F4895134F182F159616E183EEAAF92ECDB8A50CEBF966F83FDE95"
03296     "ED77F8B7E8F575B39413E1FC81B936F85FF47563152ECE2CB030F27C6E4823E3304001807F"
03297     "86CC7E034EF76DB7E56E1417D67A716E17411E1FAC556AF7104808185236262053DA7F79E7"
03298     "61CB2EA91FD4ED0295B8EF18112BD02D02058E960C70EFFDCE0BEA96614F36C0B664BE8DFB"
03299     "6F72CF9143412224AEB6885808AD2BCA178456CDA74FC169117951DBACD567B3A7C928D6F2"
03300     "CFE85A1BF6107F110089BD46C92D9D9F9DB263E468E05619F34779A32F9C19593B9765FF01"
03301     "FDB2B6ADF4C12E61C867C5D43AF7CF5E35F3DE7A603AF94FE779C8F56DEC5ADC365AAC7072"
03302     "E226971B8DFD3E7199302C4F29D96DE4FD1C5536BCF67D38136CB4F6DB7200A17ACD8F97A8"
03303     "DEEC6B2807464042EDDF6B2C52A9C709411CEACBECD980C93313D0AE88F123AF3FC456B9F9"
03304     "E31077DAB6B371A49437AE46A28E596D449D36F8F0E999854DF6562C01FE5DDFF830439B17"
03305     "44963A64F3FA4F6B0936C8B64F15549FAC43682D60A6EFBDC56540E1A99359BAF03F1B32DD"
03306     "C3AD66AB93E23AD8FF432C62BCB74E7077D629CE7B203D0DC73F1FCC4A5F40FA3D810C6F23"
03307     "7D99193D3A4C8B6EFB0ED1A9EEE9A1F4888224037B467BD3E74ED7152BD568AB0FA9DA697E"
03308     "07C4C1BD2A11FC12B3094DF1AB60644FEC4B5C6DD59BBC29B557896749107D0F4FE9532D15"
03309     "F310BC31F24E536075E2B5E83398A11D1601DB530D87B74A118011CF2E319A86CD9E918966"
03310     "25AE0912E583A72C41CC82B45BE96148DF02F48ADA4F82345E78DD20D0159A5666F10658D6"
03311     "A8AF2E72F38936A2DE379503660B8C050D763046B068B730434E1429DA842D8F777D568549"
03312     "49D0B5DAACF3B1CC6018371B2EB2F75B316809FAAED5FDE42F360B7FE1AF3397DB7CE28918"
03313     "912085C9177F0368C8DC563CACC0425F5BA0EDE0E64B15F5B5DADE1B1435AA1475EB955733"
03314     "920F2951C5A5F91EA72657EF19EB88FD3911DF4F2B4E6C5866199761B0BAE0B24461E00FFC"
03315     "EF3494037ED568697D614CD95EC8EFCA264C5DB140E810609BFC85131AF1D89429228A1E2F"
03316     "0F7A17C8AE41903A5A891B1AB23859B083A6AFF4C9F66C31634F38515214180F41BEE63D11"
03317     "AF8F7937C4B4E5F02F99D5DB9D0B42483E7C3BFEAC",  // PRN 43
03318     "8C46BE4E4DBD26FBF3C80BD9C9BF8BBF97FF93DE3C10CF5C04CA36180007A32DCA41AB460C"
03319     "E2A0604107B5442819D5159AAC554BB2EBF6807D877F0AC3FCD0C63EDE4A3DD19205710F0A"
03320     "5A7B1D4071B3406868E7887A2CF58C5FD38B5D72C534CFD40B5D307C7BD5C626ABD64A07B5"
03321     "2C7DE84189446205689744A56327DA0F4F70010A30D6CB22014FFDDD3780F5BBE687277528"
03322     "5CA133E8FF1566C8526B66F0C6CE2D4A8D3A0183513A69B412AF28ED9D7AFA34F1FF5682B4"
03323     "387FD796BFE783D6FC813CB34D2426304F67704624FEAA4CBEFFAF09D4E5A8A4E0E7ADABAB"
03324     "633503193A2A61DE0BF49786D8A29032B5632036AF6267E9A5D8712D0F64088C0F5F05EE6C"
03325     "226D6A17C6EBE74A1D8412FC4677FE2C37F50CB86C79CD5FB402B3195C4AB474F4F2C3FE54"
03326     "07D42ADC4197D752ADE718A88332A7EB0677407AC90983DF6BAC7F92D95CF0D0B5E1120316"
03327     "145F916E3AA920CCA27FD9A32AADDBFEF0B89D9472F09DF98E961132BE9B2EA7BB77FFAD47"
03328     "2097A25EB7D9B0682E954F24C7EC850AA24C30B08FABAA50048E6B66AFBE1C0D4A7CD07CE3"
03329     "A08AAD56715D9C2A5B5851236AE1A495F8E598EDC3F4CCEE03AC2C3AFB9122CB8C114E09A9"
03330     "E7CF326DC58BD4AAEDD68E7BEFF06AAFAC5A46AEFE31F3C0B8CF3791B6244DA44F783BFD83"
03331     "DF96F29BAC2FD07C15F7C845575F1984D375E52D3D560AD4EC430ACF108810665205FD7CBA"
03332     "911E95A686B49242E2B4C5BFC6307A00E1584840103CB1F8A22943E4782CCC4B0DB04AFF8C"
03333     "E6FAD12FBD844E365EE59C6302889081CED8D7528FCF3A1223BA6554448356297182 6E7017"
03334     "A59D0B7E5171CDADF4F02FDB3B5C2E1E455A44FA8ADD4791BACDE5466F8BCBD584D09E0041"
03335     "9FB4EFB0E01C8EC5C93D065F3DCA4DE1D1A50CBFCE4D7ACE0567FD39AABBAFDD35FCDCDE20"
03336     "9EE087B2FB6C474546B5EFE04C1ABF4049789DF1F2AC8DBC98497D87202AABC1287CDC2479"
03337     "262ECE3F874DD219CE98D4CA01EBDE53EABD1B19062959DA6CA380E4DDCB7B7B0E1FE8168A"
03338     "A4F0CFFFCF761A3D989961933AF629DDA1F6CC20D27B6A181F615CF9C742B26234C1CF5E73"
03339     "664BE0E373380BFDC39894D85602E6EDD450BE5DB48E8C49A97D5796F046398E434D5CB5A2"
03340     "9C25BCE7BBE8ECA805279570C229F7723CD41B9FA497645CF69BC102423153E2D37410DF8B"
03341     "077BA65E45250D15E4CAED4EB21AA4F424849CB3B2D1756CC1766141CC8EB5D37BD68A0B20"
03342     "7BDFE38705E16628B380DAD8109F85530435E2BFD30CCACF09EE506EA31C81291A01D52B1A"
03343     "1BCAE835681E8FE2465BE8003A5AA8065B1148776D4DD9A6145B97BEB8243DB4CD771DD3B2"
03344     "CA51558DF738B9D2704280A78F8D5BAF6F8053D75151358EA46936A0F1F5F1F63B2D3996A8"
03345     "19798313DCD2C2C88F3E40B74D23C3396487A8DFF856B74230A81EDCC41FAED8C8FA8FE5DB"
03346     "1BC92435D8AD4FB7B82D5E7089F6B1D204D7D4974AC5C6DEB932782145F3824DAAF24DFB4F"
03347     "29F0A5AEB743A22557FBB014C1EAB0FBF3E0F9C1E9F91B86E6EE1D83152C0820F2CD724352"
03348     "2FB85CD0EC1CE4AFF8D35669D3E07B75DDE2344CEB07F29FB98E70A56BA7A70A2FBA7C9717"
03349     "592E7B971DE67ACA1D37668471D74FFB60E30A027E93F40D5B08CD65265AFD53E371FF7BB2"
03350     "2007441E1B10BE2A286A0B71DA6C7A3375383266C2AF05543C0C13686DB9515D43004236BF"
03351     "04A117FEF8B310082F6DA9C9CDF0AFB959A1D8193A48CD3ABF8D15DB1E981DFDC9CABFBC61"
03352     "68A8387486F1E14F9119C6B6C82E996694ACE70AD4",  // PRN 44
03353     "DF0B035CF6B45E44E8326F7FB58EBA2CBCB8C319E312F7898048794B18C591AFE9BF56845A"
03354     "8B4AFB1A3A72C1D13680C786CED126F0806EB9ED73256ABED5A881665FB95A2B53D97AF512"
03355     "DDBAE79121E071F4C79761C1EB98BA53DA1CEF51BFABCBDE43F223BC12ABB3E301B4A2C712"
03356     "47813A1818B8FDE9ADFCB7F30D0F5A88CCBBB374734593C0DBFABE04182D64DC13C2CCC7AE"
03357     "5C49FCCBA350C889AC9920911658AE7EBE24096F831FD9F546A32EFE18E20FCB39AFBC7A0B2"
03358     "0566CBBE4CC58A6FD2246858002346 93FFCEEB697EA88197468A826C95512DADCAE58DB45B"
03359     "4F3C6CE862C73F5CF151A044471CE862A3A1C46B71252FE40E740FC4627C9803B6996F99DB"
```

```
03360     "2AC44B4A84041DB5D1510D14A00A3F9B53F1C8E54A7DFEC5DE283D1DA9837B2C4B117AE165"
03361     "BA63EBA2F201D84274C0F3158105F6583D440B0D05E28CE96821DF99863E00DA6624A8EDA6"
03362     "1F7CD223FB9A260E3A03BA2F8263082442395F67616DCD8BA7BB294A0642CF1FA307F91BB8"
03363     "CD5A0C2159F513F99FBFF776C0A57827F91D5AD076E0144B7D4A109A01C6303A35536D61DC"
03364     "9E4B6E55E437E3DB7F94B26A7087EB21AB334F565624C1079EB1EB4741C5E0D8168327FC3A"
03365     "D3800DD8E2B93262DA4EEAD217F6E2409F7218B0AC2636655E104E4AC5B791245391596124"
03366     "E9532C9ED54155C0C9DAE8F79B28BE00229F3C4763BF9F8F075B728F8A123FBFA4F3A61B86"
03367     "0367EC6D797F8546D6ECB449F638B84E42C67D5D1112AEDE51DD95A0EF3C575D8523586A19"
03368     "186BBE9CD40A3EA40684E60BC1CEF69343875D6DA640BA068CF49C31661E01994373E80498"
03369     "FB1075ECDFCF4A7C8279B40C227192043BDF84071A36159EB51CADDF2AB9229984F186B1D6"
03370     "97E58D963532F2CAE870C8E80CB1A985DD277DFC086AB82D9AA266CFAB2784CF1A1B3AF69E"
03371     "09BDA9F0E493EF7B9038AF621C41C8F083CA6012BC74F02F92169CD7F779F13279D0EF7840"
03372     "A9D2FB2D0254744325030C571BEEB83500733F04F88626B7E0DD91CABECC3544389807 70DD"
03373     "DAD7B76E480A50EFADA5780CB2D88B4B5167754C464CAF45BAC5B2E33221837AF9777AC4F1"
03374     "83E6B85314B6ADB066405B7F6881148D6F2A6368DDE833F8AD5983BFCDF219FD1E8455BEB8"
03375     "0DAD449AD0A1C8DE947636C7A9A9049666C8470B1730FFA57330536C75B9BB13E0F6B081AB"
03376     "45B2860BC2464FA8A30EC40F27871C492EF5E1654BF66692B8D6F17407506675E8EA8318F0"
03377     "9EF95DC687329D8283D5219AEFE922545502BB4AC45A43AC9D8CEA133F92E97136CEE97F30"
03378     "C773261933CC455F3F8F1FA45FA6AC803754625455E3BE824C5640337B06B06F258E687D77"
03379     "56654FCD6226A1B2FAACB4CE6A92372BD41DC5F6DD0E3C11506629FDE1A730EC637253208E"
03380     "AF7E2DA5E56CAA14C13720C22277E1A3F12C46907215BF707B5C72EA4BB6E4EA74C9AFFDE9"
03381     "2C1CF5C8B7C77DF3C876D18338D38DB7BAE231D51F4D21ABBD09A4670ABF92E0EA885EDBAF"
03382     "58BCE7BECCDC258FC2D43C9D1788FD4123402C075AD5668E8618D20AB729D43400BA877A98"
03383     "20F5BCFB5B497482F9792B897C35E8B9D1A616AA5F11D730F0B50E5B48FC20BFB2FA82E6F1"
03384     "E57E725D96C95CBD8B277E3224C4C2274A01A312B738477BDD172DC577FE8823BF3C171629"
03385     "10557D4738D8151AFE31940E3382C27E7A1EBDB3A9207287DF6C5793D0C747B13BDBB205BE"
03386     "0422B723782C59B73355D6B30FCDA9AD1E2002205D7BDB747F3AF27F0745A4B0454CEDE995"
03387     "AE60F7AF99CFD8CFE91B6D4FC9E7CD5482618760B0",  // PRN 45
03388     "E9A5B2A753003C06D6D576D66B52988839CDC8A608EDC4DDFFEBE0BC1E1801BCF8CBCE9494"
03389     "E2E2512229A5E6EC547F4962B217D02F859477B54F660E217B5C89BB1975B6E67E0649592E"
03390     "05B4D1110C9DFA34ECD8772A515896DED72BEB9DA4A1AA2FCDC080875DC179E40A78ECB253"
03391     "8F258EF9FA5B9A6F4D9D77E9D6F0EDBE75440BC6FD03576A3AB2A323EC980999BA4129E956"
03392     "31AEED2451E9283971659E6562A45579B22A5CC67CDB4AC6CB8747672631B3E8C41590E085"
03393     "A312BCA3BA2CD9679C4E21FFD66F97AF011A77A2E5885FF6D42D6276B6EDE10CB29CC5EB4D"
03394     "D8A7110F0F09702EF12205CE35803375DBB421D167D65F508CBF3E1AFC6F5FA8D330143F8E"
03395     "45682AAB866E750AB68AD4793CF8767EB4DE37C9166D2FA419F0FD3EA0B9F5549F657E5919"
03396     "21AC38FBE05D9A2E2EE2BF5FD7520C1D8389AD34C7ABED6C8182D2801E7435A44ED80ED3A6"
03397     "4DC1E9C66F105350FAC87C36505E7211B873659D7E2817ADC8E74DFCF62B3508A0F1BAB6F0"
03398     "F16549033CE1D85CAFBF731406F9596271860E6986DC917C4796D57590560233F05C75FED7"
03399     "1034783A3C319377372B2DF2D27DB655123B30A8F6DF9B329CB8AA2A9F06CA13E10E1C9DCC"
03400     "9FA9725DA1E47F0917085A1F6710DDB04B380CD6354D980A3C64265F5C2C3A1CC3E6A3DCD9"
03401     "5F62D4DA6CCD48F83500F74A0275469DC0D921B648F74B7F612552350049FDF94141F6C493"
03402     "75F17FD512AA0982B5DA6E9F9C332EFB2007E367882439BD6F416FAF26BEBAB6F1F6FDF0E7"
03403     "DC7B2DCC70FCEE25CBF3C22FEDA4B0B0A272F049379AFF4002293232A98F69E83121269AF7"
03404     "1D06DBA5671169DF46B7BE8D5AF09920BA89998A10C30A8E101AFBA9C84D0B5C73AA622D99"
03405     "6E030FD6E21D8984723C38CC43F23D1F242176B3E6C2797A77A99B15373AC0127DCF67ACBB"
03406     "87EC231912E1D6DAF08B93934CEABA17BB7B0F599681177C4C870B27E5604026D15C0DC1DC"
03407     "38B2CC733558FE744AA0E981A0F8C8E569716B7C99040D0F317A05B70CDEDB32601E360CFB"
03408     "A5078B133C4B1BEBC4C405A66A425355EBBF93B5E53D97603B100FB3B84BD5D5404953C8D1"
03409     "4AC969E3CE880898ACB097FF0849CBDA28DFDEC014F9F2969297472B3C64E12A8B8D039D25"
03410     "D9BDABEBC99CF89C2177DE8B2A28C1A76E89A975099AAB1B41170AE54352F990B7EC7C305F"
03411     "64315D35E4CE58D8F0C71BDBC0AF217EAE5B3C330B42603B4F544E22486057656370534066"
03412     "23DDB03D7FBFF39C55A022632873D314722897F65D841BECFF55BE5D08E242FB928DCFD1FD"
03413     "1D6B599BB28AD48AFEFE2805E9CA6DC8D230ED76F920E7EC31A9E7899A328CF51ADBC640A8"
03414     "19E8CEDE8CD359E94B4FDB0EA6963525576BC78BFBF24CF0AD8D2109F9F62EEB0B97CFECD7"
03415     "947CF23B750BDCF1F705398B75A4FA80CE403A6894145F7AD4B875CF1B7EBFDFF5603310A8"
03416     "A68922074586DE38824BE987A1B7A7EF61EA41AF880F0786D00E3C87C181C8494033EDD108"
03417     "FA97A72C10DF99BB20D7D214CE6E87428FF896CF30740E7B968C40B59D4759CEF0F265DC0C"
03418     "DEE77395B2584DAFBD202E9B7D4E88BF1B7846AC11AD416A9F4C109A73BCCE420C8D56C78A"
03419     "7D3BC3140749E2EC85AC4E1C3C202A0F2249038540E8451677B396A0A2A493BD30530E81E7"
03420     "129229A3F1E8CCC556F701F8E4C114D5E036A65E83C5198CCA79E8D2D1B4619B94C7F65B79"
03421     "0DFF724F4961EB21BFE5D2147B39148677EC43D12B0A03D8F1D6C8959458B51202A19A35F2"
03422     "F99D700A88D7ED3A27162E2C9452E5E25310CCE0B8",  // PRN 46
03423     "B0E553DEDACE943FC243EAEA373E685435F5CBFD771B220AC01CF6EDD6C094E0D1C86A6A02"
03424     "4F4AE412550039E1E89D749CE52BC6343A03A93BD0F96EC0FA963C84E1E7D097DA8B7A7EF4"
03425     "2C5FBACCC675FCC0225694948C72F7D71A62DF676D3B8DB0175319F439DAB5D65E84EFF624"
03426     "16B1A1F6D35078BDBEF2CB046F3903FE7B30BA88AE983585380E146132F61F09E327123A94"
03427     "495B81917E9DE29484E9A2452FE9429549517175E292767A2E78A2ECA42B9FD03F0D9481A2"
03428     "F2DDA62D3ABBD6F4E5B44ADD9DE7DA244553FEE82F7717410D99A54EFFC1D1B186FD66D844"
03429     "9E08D4109BCEB024A685797CA4CE98ADC2CCE2EB207919EFC5F8615914D41B6538BC82E183"
03430     "7A2ACAD54E1BDC0D316270359E6102D8B4CEBB91DDE907150A9A6A00A72953B3C4FDC3AB2B"
03431     "27ED005A034E7B5FE10072A4D79C1994947A5E2098E043FBA08D2D54011F73040FF023687E"
03432     "28989737C87C6D997FE78C9E3BF7DF6810A57C3362EFE9B741A5C12FDB5EF8E4EE7CB805C3"
03433     "84025B0DC3B667EA69807B83377AFE7DEAC5F9DAAD6382C00F556B50F94C9C3F3F9862AF9C"
03434     "687CA48C762A71D9E5F95AFFA8CC2ED974E06A65B7AD4D9EC48B6D79B9D4267432915D64FC"
03435     "C2141CC7155720D1951A300BF9CACD89E5DC8853B8055C7602D182CD676492CD185363A52B"
03436     "726CFA90351E8504F7D53E7D19B132E55918719958D62FEF4BE6E3945EEA1FEDE18B3818CD"
03437     "F0C9C13E6E39807D890A268579DC47286999E7EE6056AAE24F800652F3A6904B508946A364"
03438     "74E08136FB57AA9401C29F1123E3849CB4156E6322F4A3A71404DB57F7338393E313D898B4"
03439     "2B244EB4FB465A4D6AB1554BBB61DFCD4EF5235D720D1083305074A5F8087C464758ED9BDB"
03440     "33CDC4C624320D4B883BB762CBDB5D92080EC3E8F5997AAD2CB54421591DC4A24778CDBDF6"
03441     "CD39D43ED7C052C3D571A5880B3E1F0EE5629F558F9AFD94B230BE3EC7586DE128849F5FEE"
03442     "E7A83ED21F9C8DCEC8BD07A2875EC0B3C94C62ECE93CAB2E3947CA74D9944CF424F5072A26"
03443     "CD57E1F065DD256FF4BF6262BB40BD5F92846929003A31902ECBA57BAECB00EBB01EB90137"
03444     "4CEB789094831A4F082D288019E40800E67FA1F8E7A7DA9277D2D0B807AFE926D0A78A5BCB"
03445     "CCCCFF63AD49212762782EDB02FAFB1F40AF5D43C4028207ED86EEB0DB39D45F08188B6C79"
03446     "E3A21ADF5109E331017739A61E102A089236C42A8BC221EB2194018697EE4B5ADE114DC36C"
```

```
03447     "001D5D21CC2678BB35FBD1B1249D0FB2B46BBEDCA7E77F8B12DA2141C8DE43AE6E0D2B7BE4"
03448     "160DE9AE81177CBBAC44D1FBA1FE14A19597EDE5D5A5ACF750A7E8DB72764958BA28DA59CF"
03449     "D2BDFB89FAB20D1D0E27C95526E1B3B9171616B0C73D0C853A9A7804ADDB3049190598099A"
03450     "45C3AC32695F0E6E355CD4565783DF2845609828BB785FA38F727506CCEC74920A43EB37C0"
03451     "BB5695C9BF02F14B8AFDC0B591A06DCEB08C446C9492028BD91A2E793B8185FC14A4816BA3"
03452     "B9E476C1F3ED607B783D4E8DE875BCB7972A2815D80EDB08D38F3D10C1EA38AC52458DFC22"
03453     "BA2226763AD3954E8698831D54F8DA6E8D78D2428F3AE407A6C42EB4BD87DBD3D10A0B95E9"
03454     "DF0F8BE9A9B26E067A1CBABE1F2694B6E3E16FEEF64D53CC599AE6A1D640A2CE921B67C182"
03455     "09DDBB2A8D74C10DD35B518D0FF7F01354D8C08A5D3FE18204BABE389BF2BA391AFD134C53"
03456     "A5212DCD30640163BF12155855848A1994AD1F9B017C4456AFD816DA9A797B7C68107E9829"
03457     "E9277E338CCD29CDFF526FE3F2FDD8BEBC40CD6C1C",  // PRN 47
03458     "07DBAC0D07BE0E218F5C403D29D1CF92F57C3C878B36FAFB77FC90EFD7C7D55BD0F6F9143A"
03459     "3FB411962F0C116EEB29065283075BB555CCE1086B8BF5D9F632976D6C764F21E4F7097E04"
03460     "290848CF54DD98EDB896405AFAD2329DF1852647BCDE28AE8B5155462CF6D85266B9A45737"
03461     "0007223AA335E790EA2796DDEA6AD2EBDB79D48970824484CF4AF8920FF7C955BE43276A20"
03462     "33D3CBD82108D5D2799186347CA33D9C2EE605FD7E04D456ACF1A2511428FABF170EB3DFE9"
03463     "DFEA5AA25A2676839B6B71378D5A16069DF73628705F9004958EA93C8BD5E595D6B69D3045"
03464     "A8DD864D29241D2969C2CEF06F2999CE68BE86734C96CD8C8F030AF8FF81D671354A179336"
03465     "FFA0400AEE9B659E2435FE17B10286365678A33704A2414D08C9D07AF9BCE755E4057893C7"
03466     "71AA76477607F5D73023841F39AFB26A57A8D6AC821AA4FD013C838E95441BBCAA3026BBEE"
03467     "ED561E166D7DF9F441ADA96C4BC0FD7EA1E4AB9FB97CF364F83E925430BEC3DFE327112CD8"
03468     "EA8863E657EACD4910FE726720D3F67E35D8C9605B19FD4D0A2443F92052A54704E15E971B"
03469     "51C50E4BE49D1D9B0DC1BEEFC3DFF1B865E757FF0B583E23848D5FE49245A94E5426AA4CBD"
03470     "D42979FA806FE21B54425937BD6C49FBAFD86292182F1BA565D3006AFB5F0CBA343163AAAF"
03471     "45D586885A105C89E2284E1F4487BA8E08F673A64F49711E14C9A6CA1A431FA812F3CAB663"
03472     "030307A41CD97AD73AFF004719E35F3300D1C4D94A0637C65A3885485C3EB6D3CBA1DEEE93"
03473     "1114C39CF2D094EFB0D9B3AA78260062DAFBBD41D30EF695816966BD0F416CED7E1951A619"
03474     "45C2DBBA6F7E1B277669A33E40DB63C07E33F49AC660D68A6E35C5142AF355D6CF23302D12"
03475     "88F2EA8BBFFA1C0946D9F9391695D752A0E10883D010E93C19FD3A2F7F11B114489057E18A"
03476     "B1BD1382FC2A3E5FA4989DEFBB888C4ECDCC84506A5CCADA0964FF8D804A9B5D04ACE81F21"
03477     "02A7DE0E6F85335E5788990374C1790982861E04836580A0BE12DE3B1E62105103BBC05FDA"
03478     "B5F0307EAF3D65B75FF39584EFDF97C95C050EA0646C48F6ED5B1C949A2145295C054B23BD"
03479     "55185D5B36710117D4E7432630AA19AAE7BEB3E694A8AEA0995CCB45A11249A9ED0326705D"
03480     "81ED456A5656E75D9997EDF60B16AA9A2D6E96067D0D3FE291C3B8B25E5C53B82863908915"
03481     "2AACD3DE1C3F177BED52DB5A71F9B89162F2026626DF62F45D9CE2D085E77616BD1987CBE3"
03482     "FBEA54786B8C15D7C2E5C65599CA47C33E7A0BD5098799C8C3B3A0B56D3F5ECC69C0B490E3"
03483     "662201CC9E1097FEED0817891C3DB24E882335D5F3FB647ABADBF8F250CA06591FD10D4F66"
03484     "591EF9145541447B49D5B6E6334E7EE0AFB414AEC813C57C34D71B8FC70E3F2DA7B3DD964F"
03485     "526D662A6AFC32E3444A6D51AC4AC6AF073A94F877007C10C7E4B4039CDC6CDC1C2BF12A6C"
03486     "A9121DD14B6681E84675E6512B353E638491C6E256346E36B0428116DA5530691276954F0B"
03487     "B164090143C8DFBE1B88348602E163BDE9D3C95583054A7FAA921D266F5BC309E627CFC2C5"
03488     "A38BF8AA1877103B8D2768E267ACA5CD148FD49F1F29F5733ED4E0DD152DE3C0062D36F716"
03489     "51EBF72F5F6D6BFCB44486FD7BB93A42EB3C29221A807E136A36DA79D87622E3EF90A7EACD"
03490     "1F3601EAA1F9CC3AD7974A876E067E21F7E38FE09738EA4E4ACFFFF03F465BDBF99A95AC32"
03491     "653BF6328752B89C07A09B1E912E6B51898AEB2E6C39EDD54AF6EEB77D68948ED1F0CE45D5"
03492     "57D30ACF77CA25C3AAFB902761CE559AD5D9F3BD50",  // PRN 48
03493     "4778E4E7E70695F1546A20995B638803A31D9030B5C9292607668EA2C86D02453F513FD4B3"
03494     "C8CEEDBBF7FDA68E8D2B0755C2BC5A8BC6B2D978957E71A3FA1A186ED80481476C86A8811CB"
03495     "613BF3BDCFC49449137FCDCB65640CE18285D642EB2C2B3C233578DCEC221624C67CEEBCD0"
03496     "F927F3DF062B7B92A69583F187EEEAD690D8C79FC60188F4AAD5D5C6DAAA10D0ED9232CA69"
03497     "5DD215C2500922969562BB35AAFF60F81BB6B187F64E9D0939C6E4AC6C0C8BEC5D71BAB85E"
03498     "AB62C634F80AF40A9BD6C134E8353E05BAA97D0547A31F64082EDE4254EE0F856B7ABFE8E1"
03499     "601DD965AE61853E65B0BAE45B0DC9859356874599C4903E65AE0737B8365ABAD24B743944"
03500     "9B382CE53AF82F654F9415427C168BC0C30A16B28E289EED8870436FAE32C57A8781967171"
03501     "690CAF0C305FF7848A59DA8A8D499292EC903AFAC56CB98848B20B5FB6B2A052999BD8783C"
03502     "65DA04EC8A7B5845EF773D37237858BEA4373B5A77F5D2C1F9AAFF6A7A9B31AFD405079D99"
03503     "3D24B8754D3974D2438599871BE8FB259CF1D20ED12116D9568EE0805E5B4150ADE95D4806"
03504     "617FF8319B1018EA256867F1FAA7CAC159DFD0C6B2F2F9B133E11D0CAAA1B92547F9CB5DBA"
03505     "E9B3C50FF3D2A21D66DF72A632562F4C58F02384254A26305D11CAABA33630FE92B511E0AB"
03506     "AF1426A21DC33B8B608FD224DAF096EC08F9F791F63569711A102BDFE7337DF5247BA1E78D"
03507     "013C103B422825F87FA144EE19A6AC4BF27F6A2A80C0ADB4BAC22FFBA91874B3F4B9C587DB"
03508     "3223E9CC6FF4815733C31C325EBE9B4A42B64A24274C5C9C06571D0C146DD7B6BB9DAFC8F7"
03509     "96E02A403A4C8E4ACB835B4CAFA5FECAF70D59F420F5D81E5674AF08F205202D759F3D1DD4"
03510     "5F355EE679F3426CF7682BC23F055F297D57C138EF3EA4A7D1EC78E19D5FEAC9061A974965"
03511     "7AD6360B6FBB0B17DA96BBE3CE3E83A65790F82CEE9B76F1E30863FC6972FCEDB23FA8378F"
03512     "19A23BC858CB880A163BDE11827D5521F5C6D1E18C855CD0A7AC4EA42BFCB1A29C027A1410"
03513     "C9185A2784B3E2E24BBC5272B37AB0879B70F2D8C3BDC63C0D1BC43FD6D6A37DC32FDDED3C"
03514     "3291390D4F17C2876D0877CC7568F5B1159C39FF678D65026B4793996B79EF80A3128C719C"
03515     "93F33665226409D3826A4B4BAB99973E8145004B5CB736198E05C249B19F7EB1C43215E4D4"
03516     "E1E96AD1A1DA6B3EBB505E3FF61E98EA79176EAF281663B96B68A83CA0058A79548B1E3B27"
03517     "3DA6F965289369DF21B3D45CA48624CBF472842EFE8EC06F69DECC427328BA713E88C51AF2"
03518     "D32BAFAC785346976CFCB268015FB58317C832A5DC130665BD30BDB31C0C74E4DC77E252D2"
03519     "812EDF4A9DCCAE0759FC945A7C4FDB1978A2BD256B1158D3C79E7DC835718C38087E842EED"
03520     "50736904A3053CAE27C10784A32E781942A541EFD9CA641364D839728A656B27D532766836"
03521     "A5C2D1A968D5C97E8770B661337B280B9E8C6AF012BC76C2D43222DA5273D4D38725387B16"
03522     "33EACBA72F75B6E6B4E7D552B774656FFDF7615D037A8ACF8F2DD845DA21C8E372F8C5BC3C"
03523     "42CBA3A189006926AD11C6539C0911AF56B13386B6F729519A5195D6AAC61CF35252956EE1"
03524     "5736673C4E19F3EC7A2D2E5743AAED65D65ED6ACFEFCB8E5B53320B7804A6187703E53E210"
03525     "59FAED278C26FF96EB4F44B1EE2B03AA37C8C0F67C822A62C7C2C8F4F35D51BA0814A70F09"
03526     "2A513C358C9EF6E946683FAA70CC88360F6A8AEE76E212627C4F110FCFE6D24A7B118658F2"
03527     "A366678ADE49C0ED529677368E3CEC7A65040F03A4",  // PRN 49
03528     "37AF4FFE6FC0F844F8BE37B431A60B3FD62FFD1E73B410D218F58B791E84A90E0AACB7F595"
03529     "6987001AF46748C6C573D3D9857FB1D18F916DF21863CAFFFD853ABE3AE024DAF4B5443C97"
03530     "F24A4FFA884D085C118ECFDD2C66A0DA98903C7CCA662790CEF0908F67CCA1EDFEE31D71D4"
03531     "7889AD299B1FE272EE0B04D8B21CE6C00F05961D69DB9F0F1FEDEF4F6065B86A59D92E697C"
03532     "8E3EF1B112C844EF77C3EDD0648E02268DE35A4951F3BE26D9E2D7CE455D92292E48987830"
03533     "170CDF2096D5B3C136A0EDD548E73AAD4A6A79523600504CF34AFB9EB85F03E79336404899"
```

```
03534     "9AF55BF3A6B347C530018F5466787848601065 35D2C73B44C4BAD39AE74426E8CA0AE666E7"
03535     "FA8C51AF6B6DCEFC4F5584FCD29960BB9298353052570364A1936E9E0118D7D6C0BA3E50AF"
03536     "8B3ED56CB535F454135ED900EF59114E10AD6E53B908F73B5ADFDDDAB01C95CBFA739B7F8A"
03537     "199714C26ADCCBFA46B96D4D4C79050408B65357241AD7BF55349F085B8C040F65287DD442"
03538     "CAD5A2A970959ED5B00AAC659BA6AF7CE5C26EBE51E4A471CAE001D74B13C71DF072E978F9"
03539     "647D75123092FA856604BE736B6D03DF1C8656746E0906FE462375759EDF12793C475446B8"
03540     "CE6E2EFF44B164C2E7477AA22F50C348A2BDDB65E6B1F021B5216897ADE7FBC70A0767E514"
03541     "9234E84FD7AF5FB9F22074129C8E4E15F935F451E36AA17F50B1BB698C302ECCFEF8B4CC22"
03542     "9ED8937CD4BC5F8876938CBAA810BCAB0751EE66DA1AA2857B54534EFA4D03D5E1F681D15A"
03543     "179BEEB40630868A2070D0C659E7EA3FD26308F5CD6D75B97803C017E18D4673F6B453ED69"
03544     "4B32083260C43C40EF81AD8CA230426AFB9020D8942A2A23D4022ED50979CF8537680D96B6"
03545     "CC165F9ADBEE540CAB59E2126FD5CE6852B8613C346BEDD11A46CCD6A04134FA298692C7CC"
03546     "B34457B8D0EA863C37383C4E990AD3918DC532FAF8B15D8F610A85B03C2CC4650FB1BB8C05"
03547     "C1C0A0AD75D2E80358D2376BD4AECDA2172A3250D59884A5728A74876F736B4D24CCD5EF1F"
03548     "833A6A4C6F2FF8648D073823DED852109B67BC9BD487E2BA991D958B2FFE34B403F355B7F9"
03549     "35CF8A63037C8C6100459863F58419847B389D92DDA586AC3A3539148E81C810FDB6AC7EE0"
03550     "E731DDBCCB9A3DC646C8EC2FD37C77D365B5E257BA519D07C4D87342117CEEF0687D686090"
03551     "FD1F002274149E3EA39A2A91B0227170A6CBCFAB70E3F74F3F8BA317A5642E8846110FF20F"
03552     "2E0DFFE5818C24432BB94A2C4AFFF87A7C72BA3E206B300ED5E5AA7624DD7BF84A52E4B7B5"
03553     "6B2353F2052883EBD1E9CB7AA55EA24EF402881E13381C0C76B35719D3D8EFED5C8B823F77"
03554     "A1A203E3DFB83E51BBF0FEE41CD5B975DAC38AF8FCC0527BE1828426F11D3ED5676897D756"
03555     "EB3FE2C34A51F6FB9136AC656893826D6881BB5A6250302934393CDAF906B2F03BD8865CF5"
03556     "BE9DCF78DFCDEC0711B6712ADFDD6A2A71FAB1D6FE40CFF4BB1267D169FAEDBCF21F1379B5"
03557     "8CC76731093BF233951FEDCA86A2BD505F9E10ACF169943F20846A341D85297DB02288BC9E"
03558     "FAA7B43691DD53CBFC170B4FD7D3E11FF76C30B922C8AA205A376F94D61066749C2C7387CE"
03559     "7EC5C6952AB7C9F12E7C11489AD879F8ED43015D1C63FEDE9EB223D52D79BE1E2242F9B92A"
03560     "5C64C8B6D5C36498A9A8F862473C081A58004ADE1B4EBD4C4C0AA5978074FF22AD2CE1A70E"
03561     "2E1945F9B14E3D537E07C930864EB4C09DDA24B74264FA45903008E42059DEFE5CA0C04D45"
03562     "A44B716EE0BE551C47CE7D23182370FB896B9AB080"  // PRN 50
03563 };
03564
03565
03566 // Galileo E5b-Q secondary codes
03567 constexpr size_t GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH = 100;
03568 constexpr char GALILEO_E5B_Q_SECONDARY_CODE[GALILEO_E5B_NUMBER_OF_CODES][101] = {
03569
      "11001111111111001000101001110111000111100011000010010011010100100100111111101010111100101110010010100",
03570
      "11111100001100010111110010011010100110111111100011000110000000111000101101011100101011011010101100 11",
03571
      "10100010111010101101011101001011011011111001100001100110111001000001010000111001001111110010001110 01",
03572
      "01110010111100101011000100011000000011111010011010111000000000101100101110000100110111111001100101 11",
03573
      "00010011111100011101011101001001110111100010100100011100100011101000010011110100001001010100110000 0010",
03574
      "01110111110000000100001000000010101110010001101100100010111000110110100110100011010010111011010001 1110",
03575
      "11111110101111000101100100101101110101111100011010011010101100010000001111010000101110110010100111 00",
03576
      "00001011010010010100000001110111111001111100011001101111101110110110001010001100101000010101001110 111",
03577
      "11011101000011100011001000011000001101111010001111010101001000010111010011011011110110101011101111 00",
03578
      "01000011110111101010100100001110101001101100010010000011111001111001100100001100001100100010001111 11",
03579
      "00000011011001101010101100110011111100000001011001111011011011110101001011110011101101011100001100 0",
03580
      "10011001110011001011101111111101010110001001001000010110010111110001100011110000110111101010100101 1101",
03581
      "10100011010001100110100100100011110011101111110111110100010100011110110000001111110011101101001000 10",
03582
      "00011010010100100111000111110010001010100110111110011010100011010111011011100111001101101111111000 0",
03583
      "00110010000001001010011010111011100100011011010010011101000110100010110100111000010101111001011000 00",
03584
      "00110010111110000011101011011101010000111011010110011001110010111111101110000110001010001110011011 011",
03585
      "00111000011100011111101100001101100010011101101110111010101010011111101011011000010011110011000001 ",
03586
      "01101010001111001011110111111111100101101011001001101000101111110000000100111011100111100011001000 101",
03587
      "00101011110011010000100110001000100110100001110101111111110000100001100111110010111011011110000111 011",
03588
      "00111110010010010100011001111111101001101010000101000000010111001100101000010110011010110111110001 100",
03589
      "01100101100011100011001101101101110011111101100110000000100111111000011011010101010010100101000000 0001",
03590
      "11101101010000101000010011110011010001010001011100001100111101110111001001101000110010000101100001 00",
03591
      "00101001111011001100111010010001000011011000001100101100101011110001010111100011110111110101110100 01",
03592
      "01000101011011001100111101111111111101001001101010011110101010000111010000111101001110000100011111 1010",
03593
      "11111011011101010111110011001001111000011000110010111100000000010101111110001101110000100101110011 1010",
03594
      "01010110100001100010001010011010100011011001100000010001001001011110001000010011010111100011111111 100",
```

```
03595
    "01110000000010100010110100110010010111101010000101001100010010110111101101111010101010000011001110000",
03596
    "00010010000100001010001100110000101101001101001110110101000001111101100001010100110010111010001111111",
03597
    "01000011100011101110010000010000101111010010111101111011011100110111011000010101010110010110111010",
03598
    "01001011100101110110010011001100010001010101101011100001111101100001111101111101101001000011001010011",
03599
    "10111111000111110100010111111101110110100011010110010100101011001111001111000100110011001000000000110",
03600
    "11011010010000100101010001000000111111101000111101101110001011000001000110111000111011000001101001000",
03601
    "11101110001011001000000001010111101001111100000101101001100110011010111110100011001111111111011010001",
03602
    "00101100100010111101011111011000001110010101110001100001110111111010100101100010010000110100100100001",
03603
    "00111001000111100100101110110110101111000100001111101001100000010101000011001101110111001010110110010",
03604
    "00111001100111110111001010101001111010101101101101000010110010010000110000111110110011110111111110000",
03605
    "10010011000000110001111111011110101001011000100011111000100011101000001110010101000100100111000011100",
03606
    "10111010100000000110000101000110001011011000011100110111000001011110100101011101010111001011001101111",
03607
    "11010010010000011000100011111000100001010100010011101011000100100000111010010110001111111110100110100",
03608
    "11010101111110110101010001011101100001000000111011000111100111000001110000010010110100100110111001010",
03609
    "00001111101001001010001000000101111100001101011101100000100010001101000010001110101011110010011001111",
03610
    "00100111001011101001000010011111101011101011110001100101001000010101111000100110001111100010010110000",
03611
    "00110011011100001111001101011010011001110100100100100010100000101000010001100101111111001000000010110",
03612
    "01010100111011111001011000010001011011010100101000001100100011011011000011100000011100010000000011111",
03613
    "11011110001101000111110001111011001001111111101011011100010010001110111100011000001001101010001010011",
03614
    "00000001101100010110111011001010011011111100001101000011101011100000100011000101101110001001010001000",
03615
    "00011000010101001101101101110100001101010000000011101110100101001101100011111100011101101000110110101",
03616
    "00101000111001000000110001101000010011001000011100110111000011001101000001011001011111111010101101000",
03617
    "01011110010000101110000011001011100010111000010010011001101010011101111001010101011110100000000110011",
03618
    "01100100001100010000101110101101100011101011010110110011011011100011100001100100011010101111000000001"};
03619
03620
03621 /** \} */
03622 /** \} */
03623 #endif  // GNSS_SDR_GALILEO_E5B_H
```

13.700 Galileo_E6.h File Reference

Defines system parameters for Galileo E6 B/C signal, as published at: European Union, E6-B/C Codes Technical Note, Issue 1, January 2019.

```
#include "gnss_frequencies.h"
#include <cstddef>
#include <cstdint>
```

Variables

- constexpr double GALILEO_E6_FREQ_HZ = FREQ6

  Galileo E6 carrier frequency [Hz].
- constexpr double GALILEO_E6_B_CODE_CHIP_RATE_CPS = 5.115e6

  Galileo E6 B code rate [chips/s].
- constexpr double GALILEO_E6_C_CODE_CHIP_RATE_CPS = 5.115e6

  Galileo E6 C code rate [chips/s].
- constexpr double GALILEO_E6_CODE_PERIOD_S = 0.001

  Galileo E6 code period [s].
- constexpr double GALILEO_E6_B_CODE_LENGTH_CHIPS = 5115.0

Galileo E6 B code length [chips].

- constexpr double GALILEO_E6_C_CODE_LENGTH_CHIPS = 5115.0

  Galileo E6 C code length [chips].

- constexpr double GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS = 100.0

  Galileo E6 C secondary code length [chips].

- constexpr uint32_t GALILEO_E6_CODE_PERIOD_MS = 1

  Galileo E& B/C code period [ms].

- constexpr int32_t GALILEO_E6_NUMBER_OF_CODES = 50
- constexpr uint32_t GALILEO_E6_OPT_ACQ_FS_SPS = 10000000
- constexpr size_t GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH = 1279
- constexpr char GALILEO_E6_B_PRIMARY_CODE [GALILEO_E6_NUMBER_OF_CODES][1280]
- constexpr size_t GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH = 1279
- constexpr char GALILEO_E6_C_PRIMARY_CODE [GALILEO_E6_NUMBER_OF_CODES][1280]
- constexpr size_t GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH = 25
- constexpr char GALILEO_E6_C_SECONDARY_CODE [GALILEO_E6_NUMBER_OF_CODES][26]

13.700.1 Detailed Description

Defines system parameters for Galileo E6 B/C signal, as published at: European Union, E6-B/C Codes Technical Note, Issue 1, January 2019.

Author

Carles Fernandez-Prades, 2020. cfernandez@cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_E6.h.

13.701 Galileo_E6.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_E6.h
00003  * \brief Defines system parameters for Galileo E6 B/C signal, as published at:
00004  * European Union, E6-B/C Codes Technical Note, Issue 1, January 2019.
00005  * \author Carles Fernandez-Prades, 2020. cfernandez@cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GALILEO_E6_H
00019 #define GNSS_SDR_GALILEO_E6_H
00020
00021 #include "gnss_frequencies.h"
00022 #include <cstddef>  // for size_t
00023 #include <cstdint>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030 constexpr double GALILEO_E6_FREQ_HZ = FREQ6;                //!< Galileo E6 carrier frequency [Hz]
00031 constexpr double GALILEO_E6_B_CODE_CHIP_RATE_CPS = 5.115e6;  //!< Galileo E6 B code rate [chips/s]
00032 constexpr double GALILEO_E6_C_CODE_CHIP_RATE_CPS = 5.115e6;  //!< Galileo E6 C code rate [chips/s]
00033 constexpr double GALILEO_E6_CODE_PERIOD_S = 0.001;           //!< Galileo E6 code period [s]
00034
00035 constexpr double GALILEO_E6_B_CODE_LENGTH_CHIPS = 5115.0;           //!< Galileo E6 B code length
      [chips]
00036 constexpr double GALILEO_E6_C_CODE_LENGTH_CHIPS = 5115.0;           //!< Galileo E6 C code length
      [chips]
```

```
00037 constexpr double GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS = 100.0;  //!< Galileo E6 C secondary code
      length [chips]
00038 constexpr uint32_t GALILEO_E6_CODE_PERIOD_MS = 1;                    //!< Galileo E& B/C code period
      [ms]
00039
00040 constexpr int32_t GALILEO_E6_NUMBER_OF_CODES = 50;
00041
00042 constexpr uint32_t GALILEO_E6_OPT_ACQ_FS_SPS = 10000000;
00043
00044 constexpr size_t GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH = 1279;
00045 constexpr char GALILEO_E6_B_PRIMARY_CODE[GALILEO_E6_NUMBER_OF_CODES][1280] = {
00046     "E6648AA5EFF0907A170377FB20CEDEE1E8D253DAC2496831010336B444276BAAB17E599548"
00047     "B1A79C67379F98DF0CB81AE8D914EE4947093ADCB94FF4B3916EE562A4CAFD4A5A04972160"
00048     "6E55FFFEB26C949D7C8B0AB7AD2F7DDBBF88A9B091510D209AAA3F6C83ECFB8DE21B77E793"
00049     "A9E759E73A7ED6330395FD9380E8E86F0A511E0FEC8E0B6501983FA7C693FFFFFF719EFE42"
00050     "49EF295B578E514817D9E86EBC10917B3B7656BEF6C9AAB8200B20DEE54BFDB18728A33934"
00051     "50F20B08A7AC84A9F945398E13434BD3076087F32CA165AFB94FEA9686924E831BA4BD76AC"
00052     "F62D7B216BAC53F71FAF1A13580BDFBB56DD555855A9E2541FFDED2ADFA6125D73F116006A"
00053     "715596001D1538B0D91C8D445D0342B910219EA64B77A7A245E4E70AD547590C3D59DF32CF"
00054     "8ACECB6D2F17ABDA7F89DB09C96A2E811599A39191CEE8F8D42A0BC2A5D00D8D7C318E8996"
00055     "470280F551D25DFAF153766F6D237236B23FBE1E74484CA9485848CD5C1966705EA03A0220"
00056     "E84B1AFC9F08F8BDC110D9597973C0F347F5259591EEEF00C399D9A6EF140B759465E07F48"
00057     "7ACE9B8CE34EB219AC63C01127520C8E2D2FA7090AC754976B8DAE5C51F18001ED095518DC"
00058     "659B5746B390226013A6DEF1C54DF4BE1E60CBAAFD10D163E937BC753184791966B930990F"
00059     "9DF0A98D2F9060233858CD699613088876272B23C6FC27288BE7D258CE3E83A0F2D5C3A9BC"
00060     "FC0E7C9F02AA0A97AFC31A286E9CF85BE5885FFA22E51827EFB1066452B27335C01FCAE645"
00061     "B56819B9D7C7014B4B8C050E647B1B215A45F8B939624433F0319C840DEA265D8631A1EE9A"
00062     "66439DA99563DB168E9C1574E2730F6AAF709AA145B75DB885B14FB78E38A05948ABC6206C"
00063     "793C33CE9189BFF19338C",  // PRN 01
00064     "A2AB1DB576F2061D6993AC3832FF041C388E44240AA6EAB61ABB450D961ED6C8088C05313E"
00065     "8A31B322CC757C389FF8B033792BFDACA5D3D4926321497DDBBD7C40F833AA87EC0D27B11E"
00066     "215E491528C0919B2E0D3106DBBA384EF22B1133538C78C76C4ABAC8493E18A30E842CD005"
00067     "01EE9A737F8AD7E7D78EF22EA826E8973BD19945A5A30A4101295D4D9010CFB232E2187A98"
00068     "AB9876DE1C9B33B23A8328028E9111A3AD612D7583B0E6A502AA3AAD7E90680013A3C2E83C"
00069     "475C687307B9B4B71B668EE7B2573756F1AB9FC7DF4DBB0577C87FEA7031A1A3B55D054731"
00070     "C1FD7849296D03F66BECD04DC25C99BFEBD810622970C160248356A15A18C3F6907E39C4BE"
00071     "82EAC8AD3874D3EC67E77FBAF6AB25DE3FE764BD22B3EC24C944E4BC060675369DE7642972"
00072     "DCA19A687F6680CF598EEF1B29BA7A4C15D43A6FE33AA867B929F158E7CDECC08E11B6E352"
00073     "52AD148ADE7DAAED8B17648FF1E98094E560B89B75F72084BCDA4B93D237555F3FA2272253"
00074     "7F4A7AEA9BEC228D7BA9DD7189B1165C35D9BCF7791E4823FD81A05131D9DDF0F696458C93"
00075     "FF255DEDAB2C4B55B64F5FF621B8A4773C1A95C0B47D84E2F410A79D8F959C76AEA247846A"
00076     "1666845653BA47CFEE762C6846EDFB5DFAE144FA31501FC18D8CB77160B8B3FA5500C30D5F"
00077     "B1C8B0B8F5060182F278BD748E3B64D8DA7FA85A48E0179CA3A2441B2205FBC5E335BC49D0"
00078     "B1FE60B2D7F94484C2921780B31B9968747EE4D19F1AEBD314F8018F47649B8EFE6D8F7230"
00079     "4E810FA22301B620674202651D73A3C3AF84CC259F26AB775F46FD947EA5818B055224667A"
00080     "94C29D0BE44E925053CB0A35FC670A1FEBF39EB00D7D64AD23AF036915DB36FAB654AD9D5F"
00081     "39B3F97F16DB0B9522718",  // PRN 02
00082     "D7C50CAAB17471A735C51D4759644BE4A788FEA8ECD5F17E00B373DBD24917481DA7418208"
00083     "04EA5CB366885ADFFC988EE21A0875EE7A36981726B7106EE3C7E4C1435890E415A2F74667"
00084     "633E1DC7D259FF2C3A1D408FEC4D1792326E7691253F6DC3E7B9B43EB546AE28E6152984FF"
00085     "677E4585B1637B8A38C7042138050888F88EA0479E974833EA5DA1B445350D4322DB625F56"
00086     "7F6DCC3F1A8F4096BE5333BF0B9C7F5545C5D7430A9935A2001FD1742AA34E572F886892AC"
00087     "061B4A1D306B91928F2A8C159E237D23A4BF9D08F6780F9936FA6A63684500CDC51ED6AD2B"
00088     "23574C459B83BA0504CEB21980F324F91A37A48E8402155CB97798F2618DC292FFABD9DB80"
00089     "00500C30D4B85F4D6249B4033D098C08316E570B10E5C939DE08C1DDE918B749332B5E6AA4"
00090     "88A5AF937F609414E885C77A953D8CA93FD382CA84752F54BD2779F460F0EDEE48C99AEA83"
00091     "7877EF01AFA9687F334218F8EDB23EBA3FC0AC8CF8BB5794562E6D5B0A356CDB6BCFC0CAC2"
00092     "ACCE6CEEAF102CA980B23A36F89C35470D0BB41C8BEB2BE0E6CD0C9689DF2D1907AFE689B4"
00093     "9C19D4539AA1A48412B6F15227839629D2A976249FF057261940C6E88719EE8672E8A772E7"
00094     "CD10DF669B1E966A8D93C4B3E9B9D4D0148BB53F272208F192B370B17291D33573D8BA7698"
00095     "4B2980343928DE4F7AD5E335FBCCFDBBD31C8E9FA63015327D0517EA2F161EF5A08BCC0856"
00096     "B6D5BD210FA3DFAE918EDD7C4E444956FFF2455A4C3C71EAB715A12FB9DF145C7FB4F46A17"
00097     "C877797EE158737A730F8937BC6DF8325815464EED4F48ADE718C01CC7BFE5979D71D1FC87"
00098     "994524F89CA789788BA057F9D499EDC7E1DC6BFF41920FC1B5C3DE263D5768B31237BBE750"
00099     "617E5802094537DC8E62C",  // PRN 03
00100     "AA696973C3A5E818309F73264D9EE219FC69692F106F7FCB8ADCC3FCE9816C691C07B996C0"
00101     "E5DEDF63501B63A6F9E5F219FCF8ACB811E980346BA432CCC6E2A0FA95B9E30ED03468AC7D"
00102     "1C0A59035359666F1A04C7673F7FB7C5576F6B2C6548979F480754D4C7C31A4A4E125514DB"
00103     "FB61BD199830855A928BA58531257E01F5D69D2222C56C0472C3354DEA19D563EFF9E25150"
00104     "256F2D7E76A0F719B63AD6917BA13C69D49B228FDE7D4181B843AE23F4D8740D942F27BB70"
00105     "3199D70A0EB5D3C0DF04F0BABA9AE21408C6070345FA82A69792BB88F1194C6C205008A729"
00106     "3C077169BBB58281CF57DF1846C976AFBE5BAAFBF05C4948AEC474CE85B0DDA450441DAC1E"
00107     "778BC63D1B75D867532FA3534CEEC0B796EADA76B8EFCC1FBB70F4902086E7C7926FF19EDB"
00108     "75FA9848525068D591D1F0E83995BC10C0A25DD8E22245AB72558D27F620E785E5132062F3"
00109     "AF642EB9C83944714CBF5A98EA18402500380FAA6EAA7823E00ADD6B70DDD1C879E486C6C3"
00110     "9FCDA5DD734798AA70ABC07AD7867F03EC3C1D478227DCC91701AEA7E509085AB319858790"
00111     "D692AC9B1A85537E7A16D4682D69E0EF5FAE63B342D86C6B610DB41022FE46A8013A756A31"
00112     "31C0731A213C3BB07981ABD53C22103C5094636CD630BAAEC9400E574DFED45FAC5E1AF832"
00113     "06E62D1486A80ADCEAC8A89D3B8F9BC8C88ED5F1EC4A31CB16487510A054CACA7725E3FFC4"
00114     "AF4C65A7DDBE4D0861F4593F84CE36C5275C543B573B3ADC1659257DE53396F667E8A3F437"
00115     "5DA4AB23EB622546547C1E51EB436DB27BC3F321B15C373DE2755F3F860A8A2CE62C90E728"
00116     "717E5E8DF09AEC67FDAC2CD84FF8CE9180B4DA29D32F52FF849F64BC65278EEAD25BB54F1B"
00117     "33A2735EE9AC5766F2E30",  // PRN 04
00118     "AA48B0CC726F7B85B9ACE4D423964A1E548D6CB78F886DE941E49360E53EAA7AD303BCF4DD"
00119     "F3E8E640DB733727082AED108C3C497644800E14AE78DF95D3214D948BEDCB3D52FEAA68FE"
00120     "F9F0A98662B390CA0E98E663EBF8B7459A8526E06F639630B3790B5195F9330FFA0BCA9C14"
00121     "120136CF11560956B598A3A656A045E254BFAB4F4C9907F82C1D2BDFF35958980D49CD3E9D"
```

```
00122     "1DE1CD86D72A35385371CDA6C3BDF42256B13EE6CB959F1CBBFE6CA7493835123141089509"
00123     "B9D8741BE692FD6980C1F72F328C670FA91882AF369D8D75F640A4AEB20923738DE0DE2FF8"
00124     "C29CA505575C59B779D32F790D7AF8DB5C5E13810EB316844D1CFAD297A041D3B9BE524836"
00125     "180E5039AEE79444CA1ED352AD4DD61F7101AE2D556CECAA7E7DF1F7BF2E88BAE57F60906C"
00126     "27C12AF03780E42C1F0F4A22229883133268987 28E5CE8E09267363575F9B2EA976495E739"
00127     "52C05E69F955C5F66937A4BE157D5BA63F529A28846FC25D800C8858F8BE4988681FDB6294"
00128     "DAD76E501C40D8A49B816AE8990D4EEE2A029DC9BA1717EB7C013ED8D6F57DEDA4784960FC"
00129     "505B91A1FE720F11B95DBF7900FDF182D0469B6764E159DCC1703CD4CE9AA2A2734D07E86B"
00130     "376A2D415C2B11FCBFFB0C249DAB0B48D27685D3CC2074E0F77AD08206DAD6BE6127A45F97"
00131     "0EBC1FED5CA772323C0D70369E539065188FEDFB2E9968425C3D5F38565380D958075B7EAB"
00132     "6FCC7464BC4A77CC23EB07E81248A5ED68AF800C5A19074DB8B0AD6403B8875CD9C9E95CA0"
00133     "9DB303DA879AB468BC213F248DC27FAF92E32F6B07E366F43A86EF30F8F3CB09CF94069E68"
00134     "546C8600964FC0DF7F413D2DFEB5CEAC12C176E49023AD74209838CF0992DD0D2679F7FED3"
00135     "BC236707A1342576A7258",  // PRN 05
00136     "C529A0B4203614FD1D8451D10DC39ADAE617BB03A918BD89FB81D4028113724B4C7E84ABB1"
00137     "977E35F632BC3FF3316A0D6CF465DCD39062FA3FEDE7F0CCD0764F1B1E2F27445282CEC660"
00138     "8F16A0F8FF994319C9C89301FF685294971382A2736062649EC280791B7724C35D14A9400A"
00139     "437B95161875143C86B51D4E0C319B822E8D91C09BD965AA7F83C0657A7DE5E346A27642D7"
00140     "D638A00CE0F0B77BB84D45C68121CA4B7F9307B267851AC06AC9F4D4CAD54FE8A6280A0C25"
00141     "29AB277A981433597F668C44961A813D6255E2BA29AEB051BF9432C4E75272F7C36A204BD2"
00142     "0BEC5111A288E8559D68F3242900FC227D83BBD201396E3710FB2A3CB3EEC4A7810A074A92"
00143     "0908A137577B3599C2B737B23D97D8015F1FFE6396C2360D46067767053C7A71BCC58264C3"
00144     "FD54BED7B5773FD727AEEA2A19F864284F1E0254A837E11C1B1FBD2CA747C4DFC2E60DD75A"
00145     "F044A971454C1227595370DEEC4A1F4DA39DF4A9D6416C3E4CC8B334F44EC3F5AE23063BB5"
00146     "0025A5A5B3C1252A979E754C40FD2E4A6B779C7B0D5CF2C0A37A569A126E085CCC007D6AB9"
00147     "1D8EDA3DDEDB1EA7D4B5B7DBFDA78F38552E663B49D4167B0F6A92203ED63CB7AD511473F1"
00148     "F6E976034C4F31AF3C07E728E5B991FA3F4FAD379A679EEEE2A73AF723623D22ACF8E07275"
00149     "89E36DECD8F8FFED7A048B911FAF616E48B7FA1829FBBF677868D901C0DA8B5D1433B3A7F9"
00150     "CCCB450812C04D280C8C2A13113B6149F54905636CDD191454FB05613D8BC8A36536E337E8"
00151     "807E0BAB2EDBB64DF986477CB4F93C91EBB9C2D85B0AEC22457CB07FC1DB98D52BB73E896F"
00152     "F59B1D5BA15ACA5CB9696E3410A6BA6E2E1D94B0DCC01D876F7442EA76DCAD9FB262EF1153"
00153     "CDD2B015F4783916C555C",  // PRN 06
00154     "F04412976580A3D5106BA358575C5549A7054E0DFB114BDED09ED2F8665801DE8F7D74E174"
00155     "D8393187850ED269E1055C9060C3441DFB91CA3CAD84A5C11FED126714C8CB9BB934D03F24"
00156     "A0350D3287A64D186FA5C945634C71F0D7A4FF5024BBDD2D79E8EB75DE7F31985585D056EE"
00157     "F1EBA9E4C100435776A7D3465C6DAE21B1D89A0FDF0264A5AB9F9C7A4418F6B5A317ED828E"
00158     "63D6AA566C511942203E62EBA4DFBCB818ED622CF03302E1B3A85C0B0F05F1CB78F9A90C77"
00159     "EF8565BF703DCF5D45C8D956765F994C8FB3243CE089E68E705F310BF9E9B81B044E307C72"
00160     "33CC1D4E3237F6AF5B1A699ABC11FC4C1DB7A998B9CB19454F5AB9ED7D3FFE5EE55A6184AC"
00161     "FA1D3A481BB74AB2E70E28809F92FA15CB99EA0AC7CBBA1385BBDA3D856E5C2C3AA379EC08"
00162     "8EAD486D782FD28C5433EC11F01CC996A2F15BF3B5B36752A188FAF37BFDC170A54411D5D0"
00163     "B36D52064D88FF4C79C1F0E2825CD6BD9C6811C0B136F9EFA727591E98E7D673AD2B81B8F7"
00164     "033235689F9EB78A4DA85D3C26B52F0B3D8227048A191B3C2805311072B556E8B8C3DD961E"
00165     "72CFB6235DC6ACA2D16441AC2151C37D9AAB1B13BDB6E4E82E8FDD47FD99E8BA085131C4CC"
00166     "5D801C6B906080326F65A5C5FCD0FC22862AF6AACB79152AEECE0DDF3D432E0D002539014C"
00167     "A8C7C782F5E402B12C34F440EC7439F8CE1682FA6C2B489C0FB8650E7FB20AF59341655229"
00168     "C450AA29B24B279F9B8B38B62FCFC166E00125D6AEE53C3CDF4CE41206D829E00BAEFA79B8"
00169     "DB51CECE4E2CB66A1E30A1C39A2C39598B8675A935BBAA4E929DF27CAA3D410BAE9B74DAC1"
00170     "435B3E86C9C27FD5112A9B9D9E7F7A65B14F08962F34013DEC906A487FC233DB8BD24DE861"
00171     "6A8EAAEA6F4BD18AAFB3C",  // PRN 07
00172     "E77B8CA081B04234CC2E148ECDDC38912DE5D2239953512C2253D3624D52BE57DD3F3813FC"
00173     "CDB94E83E074CC6023993704D862A4238F377C5EE88DF60F41D9F3952FC85CFEDC500A75B7"
00174     "116E85C30D978B5396FF09A922F9F065ED050BA7EA3FAF59590EA167A10270595C08F5F60D"
00175     "34F232DAAEFF2E4D9E0BF50568E619A29175BEEB97A17CFA4D733AC70269F76219E480162E"
00176     "E76304E0D07FCF1033829C13D29322619ABD732CA49B9ADD983725E2B104C15DC495E01F9A"
00177     "F7A398DB00497A9ABF8E2698D4360C37364D752403B0B625AF7B5988D7BBC11ED37BFEF087"
00178     "D9BED79958FB2699B5382D318AE8AEDB97476AA69E56FCAE2676F0F3182FC2CD8959EE972A"
00179     "99D54BD3336DAC904C828EA8C421C6557B4504D655CE7543112C43554C472B50263B519D59"
00180     "F5515766CFA480444D6C1DCB15A68386D4D518154B639F2D5E01E086EDF1A73A0005AD91AE"
00181     "618B7B05ED0C3760FDFF522593F169CEBCFFA228BB960350E3BF67201FBD21C8A2B82328DC"
00182     "E6C71912BFF568097730709C54F271D3201AC563443361D55A2A3E989725EB7C510823A2D4"
00183     "F72FFEFEC81C30012FE8EEE479CE1F53E1A12D15D2BC51AEC23D5C86F0AF59186691022FA6"
00184     "74E5BF405AA96A7E00398BDF02E4F9E66852C44E5E7D583A7D445CBDC54A6470C083C44340"
00185     "A65F6F7407A4C7BB784EB4D09A68FD41E27568294B2D13DE5C2AB7D4C7811FCFF67A97842B"
00186     "71219D9F3AD9899E8E334960DB56596A41C3645FAB842FC1CF41E8997CEF1277D961620C8A"
00187     "883DB761F1A126ADCA93A417230A419C00E4F7BF7FB3175AEB1EF614BDDF11AD3463715F39"
00188     "A61B1B001D5B9D4D5FD356F2E94E769D0D23DCB46B17E5DCD091AFA2CE2C2E819626F01A14"
00189     "CF5473A1389AB840F9EE4",  // PRN 08
00190     "BB25FD99C731209E762D4328C053ECDBD00118201FF7A8836F8E7651EF75F52CD8FD48662D"
00191     "53F8CDD9AA770C4CFD0B2FBFA07E59AE1BA177684026A24E4D96BBA3B856E56103752A11C3"
00192     "7DACB4426213D5EE47FC82C0DE70BC4973FA024DF51DF567659E6947CF51D39CD21CF7B891"
00193     "19172653543FE54F1C57232D3ADCB347691B425D566996F5F8DB09C31E309923AC3D5422D3"
00194     "760F1F22B9EF7403A7FA6F30A611CEE87B7C46DC839AFE30BEF927AC6F40478AB4084C3A5E"
00195     "0864BFC3625EB494B2FEABD0A0837A3394DC9D21D03512FF94E260665540C6AEAE2C535295"
00196     "FD4BA4ADDF9F34C1EFC4807E65D599498468F539FD0C0008AF77B926244C3AAB5857B9F0E2"
00197     "510856FC371210BE0D054D96A87B57480EC43BE4AD421FE0E00A01AAD06422990312B86DCA"
00198     "F2E26F2327CD33DCB057FD4CDF94F83B578510C4D7D5660A864C6CB1102B7CCD3C9F43F586"
00199     "2443B9034B58F1DA347749369C97DBE6FEE8EF009328254D5D66FB9F45522354ABC7A42C22"
00200     "28EE53C41C7368C7FB798000D79F0C593F1CF1E6389F1F3F973354DCC74ECDC9D4299D2CCD"
00201     "3EAE1320AFE5BB8027641B13245D7149B6DD1EE8E10B828B6F2C792C597CBCF2985BCD6304"
00202     "8EF90154A509D352B9AD20259EC7551572CA9665F7B3A96F19AF496BCCDF42FEB6340DCE0A"
00203     "27A4CEFD4755E71751DC9C5D6C95428EA2B7883E9937851962E3C6D817CAA8EC566F93EDF3"
00204     "98CBC117B9372A01B2519225356A8DA72CD6B515D8DB5DEC3EAEC7365650114E82FBD84686"
00205     "EB11DA3F5E9671471C844988375E4105C37E00CBA220A3B1E27A049B9753D064E21E79513F"
00206     "892FB5620AC48E3A0070FEF89BE9333C52E6C151FC3CE36C21155007D1B67CDBCCB4E60CFD"
00207     "3C2C59A81330B720C3D58",  // PRN 09
00208     "946B6A0BECB29F278563193DCDDABADDEF407C8789437694F61A4178F85002A7D0A5E67B3E"
```

```
00209     "EE8B911F3DAE56EF335D953FBB02C56BEF30D21C00594968C5EB59C09D778C637D1283171B"
00210     "39026884B8412E2AC160D222576490B5E901B7DFCE1FB46D6166968079E9695A8B9EDDBF71"
00211     "ACEA00BF74CD6B0F3BB0854B9BC014BAE5D6B5F51960D4768FAEE75FBE96CEDB25F2E3D502"
00212     "11393659553850CFEC40D40BF5C2FFB8110796D9C8B1E73DDFD086D289B4028D926462D585"
00213     "01194D092E0C6A4100B46320158B7462615EE9CE6430FC9D412BC3659C04DFF276FEF903EC"
00214     "7C7F847CF7F23CEEE3760018FCC0E27517F80EE5A1565F7CA3CF54B18BE0222EFE2438F306"
00215     "B6DE2EF6C2771364083338CD76C74DC0C98FFAB349B1C9E7DA35751676A390445131AE564C"
00216     "9A7EE77EFAC072931463EF926DA482ABBD23E0172343D595F7299B17633A344F652E09BBA"
00217     "A2378EC84B849DCBC21D64CDECE910779E1D4763CA753D347AC02D270983962F17BF8EE4BE"
00218     "62AE3F92C9831CFF23D44A83DA6BDD85955D1101D93EBB481593450F23C8DE87A2B1201616"
00219     "E86E05C835B4926DB6B6A90EDE6178071CFABA3C7CF19B4188F0C679CE4EB16B26DCB1E665"
00220     "998691A079B57772BD72E9E44A70F28C79D9ABB42250AA4116C75861B7F8AB411B9DB6AF52"
00221     "DF249551708742504 50DF1C31782F002AD88747AFFD2FF542139E081A300DBD69B2B775798"
00222     "6D6DCE8D349892B1528922BA05F25AB4CDDA0EA8CC3600BB3704363B5A20AA02AF3A3F703B"
00223     "2284D5548F0019CA0C7C4E7180D0A5E7BFEEB3390EE63692BC66BA8A7BCCC502C06D953301"
00224     "6273543991B2B37B1ADBAF15DACD696F427F47AF62E3F72791951DB035A4D0B7BD3C8F1753"
00225     "B85E3B45C177440ED15A0",  // PRN 10
00226     "F28C3E75420656B86725900CE0B35CA3246A13D45516426E9FBCBC7451AA52574E14177063"
00227     "D74AC2E20CC7C1F4FB895D5AD29118DD3748E106A9C7D7CB859A32887E80D842F92CE8B3C4"
00228     "E285521358848EEDFC478872A42150DA67328949408855488BC9CDFD3B50A89DA0243E6C01"
00229     "5321AC375063B36BC648C8CFA3D1BC3E01BF50F6FD3BA35D725E5B05A308C577296ACB7571"
00230     "DAD976027BF602FE4C3446DFED2E82A086BD132FF872E269F6770F0FA7E36ACC56F7DA2E8A"
00231     "5044319C2B8CE0999510F1C467EAE4BA80E86EFE289D929953E561BBC052CA5B8AD2F5B8CF"
00232     "E1669466BFE2FF823A66ADCB01819FE7977AC2F0DC7F2262411E3A790B5318FEF83A8BCF59"
00233     "29063928C1AEDE28877A6374C668A0A2E54DEA5AF1730277AE3FF4BAC240E0CA97AF936232"
00234     "8D45AE4BD78866D3A2756170FE9DF751F4827684A43030ED6E59587D242463CABF5F9404AA"
00235     "6565F305B6CEE1CF4C6F979602801530E1760E39BE3102184C98E633EB4EFC86E479F6D2BD"
00236     "6E16EDF573A7CDCCCDF005C993C5B35D6517A5E703C57037656D1CE0B66835ADDA24684FBC"
00237     "9DA33002DF1FF7A46BE81CAB3ED4EE9C5C9EE0C61CDF473D500AB3F00CB2B1BC37DBEC7550"
00238     "E5D604A8A58BF8770643FE8E165268B0F3B7CC015D2CA76B3230D14D48545F419B5AAB0C9D"
00239     "0E230FD44618D5A49C5749C7BED668C7336806758B629D823F16894FAF83160854A88DB6BF5"
00240     "19AFF6E19412A767FF1B31ACC3CE1327E97B47AAE48E368EE95E1D6EF341968883CD2E4885"
00241     "DFBEBB7D1E8FB51CD20E5A62B321F0110AB1738F31CE6797114340AE65CC0EDE548D0220A2"
00242     "81F4B2BDB733DEF7D792F401E3393E2BAD9C279ACC051AC1ECD1819DFDB55603E6AD7146A4"
00243     "D233DF28D3A4D85C16DE4",  // PRN 11
00244     "83E25EB35FEA9489F9460F1074A44727711FDF5BB180D3FFD3EEC59A3B6956D8F262BC9471"
00245     "D48479FD7DF6119A0B3DAEF645168C4AD4A70BAD6A31CE0AECCD882EC41E52CAB2EBD3C3E1"
00246     "24BA6DAA703172AA6BCDB685D365E4337F07594B1988F67D0B2B48C244D691415CB50FA2E4"
00247     "F32805CD12DC96358992BAAA34AAAF11F77CF3BC6F8EC079CE00027235A9518AF9FA168809"
00248     "4F94A3685DC578B5C4F291B86B915012F16E35E4CBA9941E6CEE2144BE736703A55704998E"
00249     "2B6850B4B6E426C4591884958CEEDE53C6C5E21A773D823AE2851C1841E2A93FD2A48E51ED"
00250     "26F00364FCB0570348219F5B528A546B00850E012E114B3663B00F5C211DF22BAA1F2EC924"
00251     "8874DA57C73F639E666DC8893FF362F89D776A1663425463FA2419583890D34387FA2B59EA"
00252     "B7FFC8365E7F6CAA7C614C283E53CDE60CE85672B66BABB95E2B74ED9EDAA8DBFF2ED92859"
00253     "0A83519ED3E66FB0601ED0211D29C8E188EA63A9D914ED789812BDD3A51F7C97C3E01EC1E4"
00254     "7D6EC642339E7EF2D4180F2E4B3DE1213FD03F7FBBE2CA7A0298195AAFE8C1E70D5548F13E"
00255     "DA2CC8788D6B1D121F577AD1CF3156E87C9D72FB97A6DD0BF3760A82883C74BBF01D856D32"
00256     "96ECB93C618CEF9304107430A24B70DFF3C6359230E9A59DA41FD62DA7A3BEFA6443F2DABB"
00257     "112540257B3FD7BFBA9AFA0B3173453FC4443D23185FE4AA8C99D1099C735F65893A75345B"
00258     "E2890A7DD39BF900BD90C634784D051CCEAC8C2383E06640EB1CD74AF88EB8FBB9494B8EB2"
00259     "00D8A197A8DBF34B3D1F4823ED06843FF370883C5843CFB9838D4CF9409E55C89F951AE2A8"
00260     "27238B2B9F72C141B659C2BD18D3CAEF1A8B7B3ECC28331A0968E749288E74F9E329D7DB7B"
00261     "6245E4600EEAE3F300A58",  // PRN 12
00262     "DBBEE74EE5F69871BD6AFDAB05095D22D9C722E36DF5651EE4DBEBA48F299D800BA9FCEA06"
00263     "6814492C5CBE2EEC67E10422F250233863E517CE10A093A44015D7557EAD2FA184A6CA933E"
00264     "F405FB67547212901078BC627B5DEC3339FEFE0C8B724D88E8A45D29CCAA238396C123DDFC"
00265     "A9FF04B21B0A0DACDE515B74B33D80AD3990B6F55069981C4026151C497473F1004B60D54A"
00266     "E2EFDC3F8C949BEB0D180601DA8DE3795034AE8DB8627F1C88A4CA0CC19FB2D6B77649FA85"
00267     "9FF190B626EB5571DD9200F58A14A56AB43A8E95ADDE4B54EAA11DBC0280067B1D3CFCC377"
00268     "B7AB21CF46C1A94C5AFEB03E1B1349CAEDDEC0616068311488A0691795324DBA11EFF29D74"
00269     "7489241741D222534DE334F0206EDD7A3593FDBB1BEBB29B0A820A349742997A3AFB3B5F26"
00270     "63EB71F47A89FC30BB759A5679982DF3DDFA12F05BFB6C73F188817CD8D5D3B3A363AB9120"
00271     "8D9A3F743CD71DBFE4618EE0D949FC4EC73E5E30C4FCE75C0938CAE89944048FFFB67C43F6"
00272     "00EAA7F5D73B0C31C48B88A1CB724A0F5F6B4D258991368C0F1D9D8B237ADC87C72DA0D2B1"
00273     "7B121559D30E3875679A812F82C386A57B53419062F2C9FA664306E36CEB512238E867C8F9"
00274     "CF9E9C2213A08B08E4A24619B0D92B5486DBEE8D2E50CD64FECB48D4025D294B59051E52F5"
00275     "95E2457120F21203CE999F8ED7F75731BFCD3E356D629FC5C98FDAAF8BF7D80C312657104D"
00276     "B3889BAFA852F8B709768E619AE13FDFBFF186CAAE895DA7E3E20A280B6C3A6B97321FEC27"
00277     "AA83BDB572B177F12FACD70A0C920791E258D81730DD705C65715E46F842DBD2FABD35E386"
00278     "166F0EAD438054D3237491AB40A40B7C864AC4E69F13BBDD254E34360E439DF0960A54271B"
00279     "A93D706653691F0239FB4",  // PRN 13
00280     "CD41661B18894EF1C3CEB0CE7B301352D1385E2C50A568EC04856ED4D4434F964A47013724"
00281     "40602E3156A45416C3F75C21C0262242FD26C878B93952350FED33239BE8D11F4E6C8576A5"
00282     "761C8A34C36F8C0F8C81FE0663DB900B49A572C9C9172ECBDD004AC4FC44257C6B3BD8DA6A"
00283     "619390C6E26AF0A0932EFEF989D3737F77E2A40C4FE221E125B63492729B77FDC493D5A4FA"
00284     "A2BD564149AF90E07497B57096F65BFB2E3A32044933C42B87B64F7D8B0A3AF0EF6694B8C3"
00285     "6EFD97C97B63175DF5110F85809E478FCC25D83D8EB303C387151E031E4FA4A4C926CE9D89"
00286     "3A301948347B191BABD5282B4A64ED6C389D1BC22D324AF8852C7956005661708A77075FDA"
00287     "82EE3206763CFFF42F74A65DCC3BC33E408D561D65964F5F2599469550A1C420BF4B46F14F"
00288     "530715EEB669EA819F5F288BE16D69F07517A9523528B1F5DB4826B11297F1EC01D933FAEA"
00289     "FE177F6ECF5E4C52DFF4F027AF5D9F91E5822B1EA23254AD3C2AA779C6DA422B976FE08144"
00290     "35F7348BEEAB926FC75690EE13CFBC421C305F565DC5E73B73B58111F1A87348E6EF9CF00D"
00291     "16FB3592B7F52B88A1FBF0115A60899ED57F8D94843EB3FC87F7AE96EBFA5B381F6AB355CD"
00292     "0C301F2B6A5B84832EA0D4E047007ED0A8648DC36E16F21E73E4A0E4992C1C2194C325646B"
00293     "CA5EC465D5EC8FF9B1FCC676E34602C271B32C4E93740E13B84C4CB3A01451E0593104EBDD"
00294     "9387ADAA37755DA90A15331263F904BA907150B52A7DEC815CE5A8E22DB7732C06C83DFCE1"
00295     "D118F6904C37211E0E56F534108D539307D44A578A821567EE4652EA0EDCFAA09EC34DF584"
```

```
00296     "977EFB3225AF28CB1F42D72E2004EBDB57F90DEA9C9887CC6EDE6214F5F8F686CEC80C46FB"
00297     "9DFD71EE4DA766EBF6F2C",  // PRN 14
00298     "CFEEA7B366F2B3687C364AF56BC702885EC5B1D0FEBC78201FE6C914CBA004720730BCF1A5"
00299     "88A73A8E31F18853DA6619DB947B00E9E4580B7925C9198F887788A986927FA2B911489016"
00300     "B959482950F61FA58844601EB766DAB5447AC006753247A4DCB3A541750AA4A6A0ADBFC7FE"
00301     "1F52EB6D2142C6EF0F65C6E99E1EC23064CA35A3DB1498C42707C0581EDED461B55FF49997"
00302     "5F889B879E4733133854B1066DA534873678C4FF8953A43C02F57D1A692E3B153FC1C0F7FC"
00303     "28C8A0EE13FDAC23BE991511B8F15ED706CBB735599F55027E89CF58F0185EE2AE805CBE75"
00304     "48D927956F4F7F5D938BF5A17220888211F2E7D759028E32CD6DE895C1E408B4727541C4DA"
00305     "51156188B6DE0376F1A3BA91F5FC1E2C3470C074E393380755AFA3F5EF8E68228CBE07CC42"
00306     "B7536307D8AF414F599F901DF5D764DC72A7EEB976CCF29AA043C5E2B203D9128ED2550B8F"
00307     "D14425E0FF4BA05BB7652C4647B2DFC9A50C20E002B0A50F5B3C6112F9BC9BA3120F2792C1"
00308     "A84415D99D8B762353838B1176EDCCA57C82E9307FBDE66AACEE5C4A2599D9BD826B5B3362"
00309     "22E8E9C9F2489356A96B0BE97F6568D383AE1B6A8C53252EE81070B86172BD2276EBABE7BB"
00310     "70ACC32D13D589821CB53A7D6EFE1437BF6CFEC2650B139F37516AE11CF67828937C0BA6C0"
00311     "9354EC70335F256B5C3B6B1C0E5C9CE4F2C53D54FB53342F7832D8C71C9A06AC78792F9EF3"
00312     "5431EE1150112979F4351FAE2F3A0CC33163571DDF57D3C8821734D779B0C5BCD7E7BCBBCD"
00313     "EAA726085271ACF8AD0B48A8BA8495A21C958C73134CDD44AC3A96F0A190BD8BC6E1444F06"
00314     "95FDDDDF34FAE8917C0101B1FEF10710DB19751FAF76E4E7DCC2A7FC69149F5D9F2C813A6E"
00315     "3D5858446149475C9709C",  // PRN 15
00316     "D1331B8F157416095A0ED294C78A4EC987D3F8596ED5B9A558DBDA2FDB262985C9380877AB"
00317     "EFECAAE078F7EA40D8682091CCE5DF5470562275B320D44CD754A08A1ECF7EED89C377D17D"
00318     "BF35D6D64B82466C93C3099B0F9C0D096B0ED064BCA7B6BD8C026728213D29C0A58CB48DD4"
00319     "6063C990CB1931C0FDD2C94C03ECFFB7668799C3B4A24B0F05AC39E787D942C2E2FCA85FD5"
00320     "95595130665439E1D2FD994ECE255349A86B8F5B3104D33EA40B236EDD75F80262A3C04A81"
00321     "F051BDBAC6C289A437BDB99287DD787E9A4282E75C82C62E9E249A8CD40217F7D036AA5AC1"
00322     "52029A1159391AFDF4703EA63CFD0DFA47A425A5CAF2121EA85DD57498677782BE381F4826"
00323     "08AD3B8D4790C3850511676007AC02AD1BB4FF01C7E87FC4F8A865A61831454D156ED442F6"
00324     "3AEFFA71BE0756987ECDD9633B711E08E526AF93E2D2918C427C58D1CA751868ACC69DC2CE"
00325     "567DE9B634046DCE84B56638B3A867ECC2A0E5E44D209100EABF083ABF96DE603C5CEA390C"
00326     "27860947FA020E6C940DE756E708A8DA50F4EAFE3F9B07D7BE8D1402101937F616E9B26BE7"
00327     "AF408020BE781D7AD1AD096B3303C62443CC3057F7FAB70490E3BD8BA8567C744942536BA9"
00328     "31D26D27EB1FC1A1DBF12AC583F01E9F721F4F77392514E63555A60371CB5B5B649C3F1282"
00329     "ECF6625B4CF566C14B5E66B57CBB5FB3DBF7984DBD34DE76FA2F5564DF43B88DC5CADC2978"
00330     "A7E29293AE2D9FA8DD53C8929BA91074D46A59EB2D88C5941DB092D1A1DDE5817C4B88CB7B9"
00331     "8F7820EFE599410A6717D248AE2E1084EAF8C67DF3E51D143CD2CCC69377C2A74544056ADF"
00332     "03AA640C2DDDBF022C419658D9CB05B2BE27FB3570B2A6B3B87799BB2620F679D230ACF39D"
00333     "319D05BF0CB7DFCF4EF74",  // PRN 16
00334     "A6AC8E0C4BFACE1FA8AAD408EE1E32A7AC3D4CAF0FB55ED10C80168306E61CC5A25713D29C"
00335     "4E7B150EBA2EFB83402D6EF7DAAD35E7196EA409FBACDC0E472A45AB74CCE67D92558B2DF6"
00336     "D3A73AEE2569E099AD0EB27CC90209448FD516C59FC5D4F6973507BCA26BB68FC997DFF7A3"
00337     "094A4E8B5ABA414C20BDF30D228467ADAC6AF196B7E5E1864BB3FF80339815E8A23818CD5E"
00338     "5AC8D72F46DC63FD8D55E0DE4FA34A7B1BA75C0B2F5F3FBD128774C810E09D52E95747CA96"
00339     "EC3C72AA44A3BC2CF4C3DDCAF857719980034D217C53DE0884AD51AA6E9A7837038640B549"
00340     "77E2328158B8E310B3F99F5A63C4785A2B5E1C86CB174E81BC594AE50F53C55802B25A5301"
00341     "6C2D96400FEBC865DE47A597A7CA01A3C13610EDB89DE57608C5824A5EE5949791ED1AFBBA"
00342     "4B92CF17B70E38C5F5C18110DA33BCDC6ED8FC23B0719013CF921EB457A3EA663598DF2256"
00343     "34A3C17DC8C4CF0F219C1202BC973A22E416B41B82880BB20BC2BDAEE1DBCF3FE3AC31AC9B"
00344     "D5817E3F0465079C3A98BF11A466FE3B206D2BFA4CA84FEE08B21F1D44C63563E278143AE4"
00345     "4D7E8FEF13F2B8CC1851E6D59BFFB38E3EE2E29D8B13DACFF3B7A86C0CE0064712D3A2770E"
00346     "3E4EBF2BCF885BB74E2DAF09A6F9A2ECA620109DF36AF492E9F1C301D3FB5AF474FFDE956D"
00347     "3717DB57852653DAC7D1A756B5908DC969143D56D315540ADF080A48816A1395E93A9E2392"
00348     "0C3BDA3CCD2A275A848EB6E2677F90AD215A04C1097A1A03AD9E2109DF4EC1BA189135FC47"
00349     "E034CC875325440D10E9C82C85ACCE3130B98E50B047985B13903283790D40DB1CCCE969BF"
00350     "64151CE530B316DE2DAF6BE27B85B3B5AC022E6BD3548498828F58B13C771DC8F76EE600B2"
00351     "DFF7D65968BFB04E65D98",  // PRN 17
00352     "927E57AF56E62F2F8AF99E1310FEB6E609A1784D6AAC0117B3C693D423BDB7E10D7710257A"
00353     "13658CC6543FF4F8B7A4CE05D9FDDC1DDA5493B9B3B74DE75C184EC37D2BEC276FF5990EBC"
00354     "0FF710C72BEDE0BFB805E3B7DDFB0AFAD6E210C0062A62A9ABB7B69F94625BFA1096F5CD3E"
00355     "52DBCFBC3268096A738D85225D8C3B49AEDC968187D266FF9B452DF79F4D2CF77659306CAF"
00356     "3EB157005096A1E94F708F88276CB80056683F7A44C3197C519B559380411A163D3A21D9BD"
00357     "EB5325AE8E0E9B3F293509D667D25C03DB6474B29C4A67D43B7F627DA75BEE37AEB2C78EB0"
00358     "D768834A3A0E5552F72621D18C446DF3174C0E6F8CB9456DF866F455263B1D7790CA6F0681"
00359     "8EC1418D675E1CD490F9E80A6FD855483F9453DFACF1ADFB77EA084E8E749678862E8E9D09"
00360     "5F885ACE09EB8B9918137ECF53C96A7E25BD8CEF5FB6FBBE81A39A445C172703D9EDA9DAD9"
00361     "2EABD3A822E16EE0C9DECDED407783B4B6B6E0130A9C67871163E73450253303E5E394906A"
00362     "86D11B70240B79A9A91EC381A4CFEC3AB81BDED8D9CD001D414B3155A7C497BF573A27E1F9"
00363     "2A7FC5206AEA7E400CB806320E103C60A9A8AF58CCD0C0EB03A4EA24CA7D106875B2A79FA3"
00364     "A8A8D1FAFA248DB4F69EC28FA632BE82D439ADB1E06804FA985F542A9852CC1522589EBF4E"
00365     "93CADC3B64B3E0D62632BE05959EEDB8E45BCB65D8729E975C8988F8001482CEF2C7AED6C1"
00366     "007BFA9331099D8335056E1B98D51411033C9E85F4999A43362A89246E0AB05D399C5B614D"
00367     "CB162BC6AFEFE14EA5640F100B2A8AA248C9413F0672B75AE08B4172CAE79AD16E43AC9F03"
00368     "A34CD16E34ADBBD9B942191B2764C9F2EF3C0518DC58DA052CA4E18E537846E50F1E088800"
00369     "54F69C3D8E0A5392E8538",  // PRN 18
00370     "999334A54881FD1A03D6C8FCC0F974F27002DB6ED452FECE0585F7E67C29FA76EDB9179141"
00371     "CEA6DE2C9E7210B6F6926E7225BFDD2E1D7FF19800B4BABF8A8C973D3531C30C7415535915"
00372     "CC5C82980CBB956A1D7965B38D2E0E61A5DC4B71256509F08BD2471A3BE363E30B9E95D1CF"
00373     "D6FF87B1D67C5BAF0EC1D0B364BCD0FC6BC3630FF030EC72DC9586DEEFDFA399866DD4ACBA"
00374     "25B118EC33D15979B2D65315FA3E9B3209681E99D8D9A74E60FBBC2578441DF89634033254"
00375     "96EF39191573786976D2461888459BFA938C553C8AF97D124621DD30851788956C43301887"
00376     "84297DC079D748468AAA119FA35606A8FF961AE3D8DC0A83CEA6F4C525A1867191042DE02D"
00377     "381EA126C5A427AD4791335AA76AA903FA4C5FA8366A7AA07AEFE023360A1AF93EC576853A"
00378     "1E75C2F5FC6EB395C65DE74B746706AAA558FB278B2641F3E2EBB912B99458D20014F588F0"
00379     "70E80A38F9A86A6202C17D7AFBB32DBAE8CCCBB6C7DA1F4ABEAA76ED0BF1A9182051A64DBE"
00380     "176D49D48ED13C0CA3D23653FEDB0F342458617C4CA7E1355AED9B173F03C503CB8AA7A96E"
00381     "215C528444357EC2BA885859A32DCBF2D6A0945D70E0C100AB2F1457AA0543757F7FEA800F"
00382     "BFF2FEF67B936CA81FC91E625D35993443E0458981D76B93D658EF83DC3AA01CF068E3AE43"
```

```
00383     "09FD7ACE741C4B3F1B3A04743B6EAD6A3DB9E7A0A231DA26844F4E1BC7A7C3D8D059AD5184"
00384     "D89625D00E9279F90A5F468AC7F76C0367D72AF1A9B3631A23C16AF444843B30612A27920F"
00385     "0AB07B6C744C86CF7ABF281B2FE35EACBD1973D3834F06CE3919E9A0605634E6CE3D2995CC"
00386     "951D550CCD38E1BF1FBC02A7D9116FDF6EEC56A4CE44C89B1EA460625596D2D13C5423362D"
00387     "BE36E6F8129E0018D6B38",  // PRN 19
00388     "81B17B29E24D537AC89F10FC1EA01C8A954A4295AFC1FF83540B172156B063675D4D67A24C"
00389     "C84993B423F481B38B227F4F054A2C0EFC519898C2EF0CC321711E5A87077586DC59120268"
00390     "A1434AEA3F4812157C0877F6928162BF9859B3E42990EEAC9CE81777379B67FD36A26EE864"
00391     "5F6BD00474F49A4E037DF6D15AA5DBA827E8E133D2C9E562FBA458AC5E08976D885AF96570"
00392     "39B9B8847ECB973B57B9CD756F290B8F82AED64BCCF4361ED525F766507DA9E874B8F80FAB"
00393     "5BEEBA885603C0162656FB691755F244BDC9D758E323101E9C6AA9A51ECB6430E64F8185D9"
00394     "33783F1B324B313A15874E70B3AF6559388189C0FBC74EFA9FCCE23E72E1E99DEC398200C7"
00395     "16937D4C4CA98C6F0E45081A5FBE3731FE5962315AD0AF432F0C8D6E8FC6B3B01E083AE1F4"
00396     "0D2AD886E15F29755CFC4EC2B6E644FB0448B2AAF86BC05966A01AC191F1D47A29DA34A77E"
00397     "11CDC472E9BEF50B4964E4D4F37B8B218549DD6ECC1FB6EDE38B45D2ADBFE534B736AA246A"
00398     "838F86CC6851F63A9D10F3369D151ABCB63FB2B519254CEA1D8A547F77CE6933A6A00F709D"
00399     "D163326E0CA45E892C8C143362F23930FC78DA9FA00472986FC268E5BF5756B1E28F0FF466"
00400     "9DC9683EE150C76E71ABC4BAEC0EC67741EAE2B973463D5B847C38997D7E22F0BD78DAC4EA"
00401     "4D0CC230F6CD6ACA4FA9B92B2E340E33C852FF0FEB83C113874A2C7FECACA4E54E818D7342"
00402     "880093F171F062F3DE7000BD475B62850BF7E6B85174FF2B1E6737B258C6A2B735137FD3C1"
00403     "B2DF9BCD5CF524B594AAAB386420FF87245E8692D6BE52701CC6AB24CC47B0647D7629FCBC"
00404     "F02D620063C14F9E6017AC95815138D75C650E391062163BA3A213C00F55AD2D418C7CAE60"
00405     "3A6B397FB20AC9830F6F0",  // PRN 20
00406     "ABFC64FF4A31070A7973A18D50A243C32E5895269854955O7EE173AA47D24B043F6310BA9F"
00407     "2F1CD9A39288CC61974379CB8DE105F84540713FC22F4E9A9888DDDF37A5BF3195E0DC5AAC"
00408     "393B61CCB343AE9E4BA94710E5498AAD096FBDB6E7CCC5571EF49499DF0CB4E5E5749A182E"
00409     "473CC085970EE388AD16A0ED63608FDB43B4742667FDF068A54EBA314FEAB2C994BBB06116"
00410     "DD3B2F0E9450085A2D14ED14DB3C6676FA8C49A8B18504279107B137B7F00D8F6DB63FB24B"
00411     "3CD97D8DCFB6A15A16F7DC58A40C9ABEA0A2A4A7F0A53A9FA116CBC1F72BBB625F58C1EF61"
00412     "1CBD8603C2788351CB9CE94DEE3B0AF607EFFA84D6BAF86744F778E51244EE174829BB5998"
00413     "87EF1F174B96AB7EAF58F2EAA99A67D4F8C219FC49972DCA210BD1173A4875F497B80B4B30"
00414     "93C801D4D0AFE4A89EE7A14618CC2F9AA743D10D0723A518DA46C42A9AD0B7C3EB2D72D2EB"
00415     "C40253860E6A08470F44680353C9AD938467C5B08687004DB4BD6FEB29BBD051747DF4B8E5"
00416     "4798076871DE9667F3EB9C8D87FEC1DB9744B1685A7D5FAA06CEAA8CEE6934C96768801D00"
00417     "CC84596E1BCA7C1DA93D9329047F54EFDB99F55DF2CB34CC3638306E193C1E33D840E22DD2"
00418     "E93C54F553A816EC643CBCA23BBFC555A6FDBF4F8D6A935772D0DD891BD80A481A8A6F2A40"
00419     "228B94BBB85DA6FEDC5E77CDF0471C58A103043AEDA666A811800C0EC300FAAACBA690E4D6"
00420     "5AA344715F667C6D607E3AFB12F7300E1B0621E1346A73F73AC0D58B5D5D8F9FE2B359224C"
00421     "0C3B0CE84F7876C38E589C285F93854C9D66D81B47BABC5FF03B4B8FFD24D71F3869753A57"
00422     "6EC8BA8DF81797283786C450528FBAEA9E84478D4E66077E3B37800508738B80DD40E657A4"
00423     "98EE423859A5182DBA698",  // PRN 21
00424     "B34C68E9590ACA5456510AE673EFC8AC2655D2BFA5EBC651AAC11A3547F47F61DF980294DC"
00425     "38FA1DCC62447EFE19A05E99B5E0CCA2C0191A554654239F665B401E3B3E7F2328B3687347"
00426     "A04CB1A5F67AD7F92FF83B3811440169O8E6AC2C5B9886EB30443B7C7ADA88A5F1B8F5B871"
00427     "C9D66D8447645C473403B0FF54861773875A52B05672CBBB396092EB4351E3F938E693C5B3"
00428     "A20FB88ECDAE480D8170C82F2279A8FAB2DAA15918A4ACA70C56B5124F12F310812F7B35A8"
00429     "AF7E9BA97523584F6EC9EE840DF2156A59E7EB42AF6CD6D2FA9C24DBFC843ACADBF061D0A7"
00430     "4582813484B6A23970A4D0DFD330C9C6D3B85B1CFB60DBF8C2E5D5AACAC2DFBB359F014E2E"
00431     "6651587A7A0D6CAC4533F5A1962DBBD2FBF8364D81764134EFC65DCEB7C59C332766F1BEB8"
00432     "31D8270A2A927F1201FED121082ADD3E4824C848C93BC1D1C8E79239C6F0C431C5526EC7D9"
00433     "3C71DF8A537C95CF583AA8020D4CDF88EA14C3FA764ABFBA553C0377F48A1F61F6B3A865E1"
00434     "D45B9734E23983EFE4BA98D9422F572DCDA2A2EE1AC8DE3ABBFC56B7636D4E874F469AB276"
00435     "9039DE7ACC7C2D0FBC12CAC79B18E159D4EF632E35546FABFCC1039BCE062CBBCDAE923914"
00436     "C783CAC9E0371A3D201B8D0FA5EF18D50138A0FE759A20597A00EC33C85506AF6195A8D8E0"
00437     "B9554058A918CFC1CA732CA943B2EE5E9DC7CD003DCE77E791CE4DBB91458404A60213EED1"
00438     "A7AA7242824D77F54642CD660DA5B92BD49BA2277A4475A190379D33F1D79EF412279F4B35"
00439     "E810C23A8F311D3787065FFC17FFA2CDE8C174476B4BF5308F799164A2E6FA1572A6E38548"
00440     "0B2E7BE152EF45AA4B6B9D8F8C30040B3FAAF202EFAA0DBA1461C11178B12806CC3FF787B6"
00441     "DFD4C426E259E48E830A0",  // PRN 22
00442     "83ED043E6D7B295FB8011BF3380B80F4DA4E93F1A605AB5461AF8642F2DA006E539582C0E9"
00443     "45B61E9FF9BF93B0DC96C05FE06632B8E996D0B2A49920A7DDAFD0D1F01612D1ED8133D76F"
00444     "4DB5FFEF461FB72A784928F87431CDEDEB4F97F3E3776FE8346D79B064C42234662116F04B"
00445     "BFB66A02964886AF42E9270303E739D12817150DA4970DB7A3C996C1E8D6A456317F7421A7"
00446     "27AC811880F1CBBAC39CF8B95B998E59BC35BBA62DB6FD860B16B1A4D716486F629736CB50"
00447     "FD124C88B5AF1149F2879F5F05F21012022F1E68ACAB1899BA6BBE41C9398A354BF21C6BC1"
00448     "EEFB0A8DC95A361EBCDAD5E878443FA259E4E8036AA7B8E46888602F4B5B72BEDA6305A8FE"
00449     "C7B6B285589E9B388116A79D84942C0CE1B4AB2EA18265DD3CC3FFD7B41C3CE9E79D9D702D"
00450     "497ECDCEEABA241FA144547469E5F8993BAD9390B57568A81ACD0F2CC4B7089F2D4C6B16DC"
00451     "590620E021DA341DBA59DC8DF95F49EE4431FD8820B591414C6A7C651832731B94DA4118C4"
00452     "198FF2AD88E7A2B6FB04F34A00197681825CE2461419B900F4E91481CC9DB65687239AD014"
00453     "29C9A3E643C156B95BADFE9BE6342AF86F5FE52F0A5835ED05097549D99C6848A925576A42"
00454     "CD54F9BE6E25D1F59DD967E989563A47E428410879EFEC2CA0EF3EFF0E45341C2B02E05C16"
00455     "760BAC6B25904D1E993509E24936BF2E5EFDC757258FC662AB3784FC24789EC5F792D8B4BA"
00456     "D2D16570FF25EB95498E56AE80C1B003CAAFC957E75E08B3BCFD39253546B186C779C41D78"
00457     "B05F25B4373A1C26F766D8C88DD7FF86BA0E2E7145C5C1341AD33816E02D37F2420FB55978"
00458     "1F2D900DFAB1F2665826FB346B9EE34607CFDCEC22AC1CB8CB5F12A88C5BEA927F083CF861"
00459     "15DBD41AFAB4D7ADB5510",  // PRN 23
00460     "9458B1CEAF59FD9721A1D5326146B5E0D33295B5A62E4D00BBAB17F5C538888B085A502BF8"
00461     "F5323B2EB8E08787EF7525F92E0CD2BB1A471BD6A868408F61775EA0A51F7B7CE488B3AD8D"
00462     "34E127D6B7A46FA74340C9D31FD72E48D42B7999B9E71DC5605599E146F494FB3EBACBEF10"
00463     "558E3C65DDB071641F72D3A02FCE12BD3B67CE1FDE526B5716C59E9C8EC73C90CAB964923B"
00464     "A3F1FAA3CF2DD0C27F8688FC515B6DB28C26C456BFDADEB92F8A0098E085B124F2EEC8B5E3"
00465     "E05379327D61D9B37752281D20F9CD1681ACB3BCE48B51F32A5E81C2BF833988F69242288F"
00466     "5AEA01911276F54E432C72351F1092A3C5748F82E106AC97D05FEF53AF26F9A8FF28933974"
00467     "A7F2B99FE7719C01D616288484DD1459B1F9BB09B8C2F2688DEF3733CF2DF9C7807DBD4411"
00468     "164EF90E88835A098C6F4346450DF340B07E59629034B13F45FA4013A5D9B2675ED60C32FF"
00469     "E809B646AF1094071E9BD7686C1F0C461011B42963670DF1593B6FDEDD2081BDF6B5758968"
```

```
00470     "83D0F1FF064F60664B14DECA1007341349790A8B47B6A8684D8F6900C15FA1A08AE80DA578"
00471     "663777A54E5ADECB315B76BA8DB4DF945176F4DBF1150D99C6CB2FB51A5D2AEA97D233C7FD"
00472     "5E80A87A767467E0012C603DA17B90CC406D8341C1D8859C2E4FDDF8BF09F76DBC389DD287"
00473     "DA04F8D3887DB917233817CE393923E9AB5BBDD72B1572E6F6CEB7E155A9F55397BA532C62"
00474     "0C35D6843EE297DF8E0DDEF71644418D673F7E688D8A043D2758D0827B0F543D5BA98AE892"
00475     "40E71CA7717EC4A6CC76E550B2746C78E055B9D324D9875CB5F6A6FA8109A98CE68D025119"
00476     "FFD817BF442F24AAA08912E195DC203084C784FE514621E5B40E4D2A6BA8420623DA289225"
00477     "84F3C75E1B48B915A0770",  // PRN 24
00478     "A38DA6E20CEC2CC1AFA8488C231069AA2E7CD81C45FEA7633EA166397BAD527ED16ED5405A"
00479     "5372344A29BDA4F3366325AA0E50777B14342857800C8B797C0984DAECA66607C52FFBAFD0"
00480     "79F2A00E71AC705AE27007F13AF8A4CF94A22A1B86902BD43117E6A397DAE0CD6C0D1C9F29"
00481     "A27A641B41ED54E65BFA43528A12BD2D6D95567EF23A8BDD26EED9E4A4C61CF5A655F1AF76"
00482     "A85993D9DA744EBBE3CDACB353F7E80C155197A4483D51076FDF207AEF957526CC0F82731E"
00483     "2AE6251875D6EC16B6A69185DF14800CDCC1B4EC9A31488C6857641955DCE7AF20112DD11B"
00484     "C4A49C37BB45C8C3AB05D09A5DF55DDDF2FAA76AE9D0F61A7A53265EB7264F37F266D5EB21"
00485     "6CBD884DAE5C119B7FEDC103A18967CBF99F9884035D545735101B13842D3AAD7738B06FC2"
00486     "CDB344896043DEFCFF0BE0F2BA9FA1898B550CC7CAD948A7825D9E6C4845E019EB086C5F19"
00487     "F52FACCDC74A3BFFF9F4B9118B400194CA10CEFC91D466CF9219BF1C9B707B915C381A80BC"
00488     "7BE858DE2CED2D22D8B2912D03043A66C902241AF16ACF7EBB829840079FE505D39BCE736B"
00489     "9E148D33927EBE9308BABA5476E38926DC0F9065899D9403D137EEE209131F65940DF42DF0"
00490     "79FD23A35C2A4BF72EACE7FB1F7A569C3C525986703410739B77487BE171AD20427634AEC3"
00491     "95FF6547B488010DF6FB60E2C878F22E9AE11B745255EC0004BB59B27C358EE369704F4EEA"
00492     "9ABEB52E452DC11AC02E0A6045F8D7F43C25956B49B8CCE2238E30CCD934C0004B5E68139D"
00493     "436887A8BBD650257870F420FD5FBE4C2A8E284DB4F7A4C658FA9607673DB86DF4C9F33954"
00494     "C1F1DAB5016DCDB2EF30E467FCC9851A597EB7CAE543FB4AE08B0F96B563ECD42CEF362140"
00495     "E330BD15627BE4AC033E8",  // PRN 25
00496     "98E4C6C9A946C0D4CACF1CA4256701ED297270D2DB732FF0310069CA84A275A64ECFAC9A1A"
00497     "441133BBEF6D3F2803E1344008489915287BD8734B20FE4380229EF469206E61EC8B2766D1"
00498     "0D910095EF0BD15F08B111A16F65EEE1252861E2C2C358BA18A8F93E6BF7774BD2AA61E4BB"
00499     "D9FA79E21A2AE8501449D2B9FB9F6033DFF636B1B09E1B8B9C4C9935C7C09D0325DF878ABA"
00500     "8672073E04A37A48A6ABE9E354429E60D3E4E022F40B78AF5EB204D32C051D6BEEAC52DA05"
00501     "12B311959F9275987CCBC81A3257C66A345D7862A0B20F6ED657F28A30CA3FFEB3BFB1545F"
00502     "6ACCBCD4AE2F89A177C4F6F60D2CB545EADF3964368AC2B9DE3B7664278363B4E06B01A9A6"
00503     "94519B9C2C3E54A93D74DE31B7D706445B36BFFBCB50CE89123B9B3DC8D5B9B7F2CF1BB64B"
00504     "0D91393682C3804DE3F44C23F2D275C03BF0D3C61617EBB755B3E104E7A7261F58D95FB64E"
00505     "9F9D2C6847A8FEB46D88BD0C38FACB70FBAB609F437F7C43E64EB8D8971EA6024E05E55048"
00506     "1504253D28E1A526C4610AAB5E480E5318A286504962404540F11AAF8F5973E8F5A57D94A8"
00507     "ED01ADCE79D9DC8CE9802C4486268A6ABFA49A69DDA19F30138DF8FF48C4A835A509E8A0B7"
00508     "A4AC8AC0B0E9181F187A55E203C3A150B623337CE0530ADDEC807961E352E1BACA97867387"
00509     "7BDE210914FA399D6C25B366728417B9FA4D72DFAAFCBFD3D24683EFCB517D11E7E0EB415A"
00510     "68236D11BB54A719C21A823D8D5D83B7168B491BDFADE5F56587B970498B3FE2861FEAC6E1"
00511     "C2294EC5E6B73F574D9610E42772E69D0741EAE23113D301BCB1C33BEE2526E87B02AA8722"
00512     "1A29F0EBE607D1DB76A18456FD0F93B6338366B7B1B36D7DE67ADE6DDF928EBFBEBB0041D7"
00513     "754C5C5D5237EB8DB8768",  // PRN 26
00514     "D50D3B57C2BBB2A77424C4879123CA284B8E9F91EC4D9760212787 56FD90DC8D6890D278AE"
00515     "BD2123FA72FF16D3A21C177C26C5702E166E12E6E70F2D310BB2961FD3FBA77F4073F4DF4E"
00516     "CA215CB3788BB645F3DF328F36A1A84FD9AA372EA282DE6EDFD22E51D74153112E440FC5AE"
00517     "DF230189DE5475D0F8E985972CBFF70959FA6A0F3F1E6DF9712A67FD0A772904675 95BC382"
00518     "C7D40D62B32D2DBB4AE5FD1E76AC30D0C18AD15C4B18058A9961B7BCB9DFABB535D207FA8C"
00519     "C51623F40B8931A21F46F4CC4D70DE064BC77663FC603CA8FCC47332D01018CC5167B6749A"
00520     "C6332C1BC19FB34A27C77DF365D218AEDC3D2E7A95980F542289E3966F9269C76D344044A2"
00521     "3CAAE9077973D71AA3348F8809F6C94A281EFA50D61B7FF4531A7FAD52268E7F788D6FD2AD"
00522     "0129E0FE8CEE9C39AC9284425ABED8396EA69C4DEB7C4EA12F24997C625465E122D2778DC7"
00523     "A8611D3794BA7A5E6BBF936A91EAFC2089C242E3866FB19E1BBF894402AC50B429E7D128A5"
00524     "64E46C63B5097EF0A594DCBD215C2AB74FBDBD1E6F3E6007238A36840DB45E61E8EC8AD9C3"
00525     "B1573A878C0197CAA649490EE9C6BF3E3D8246FFC7D7A2291FC1E7592D066C40C74463D9FD"
00526     "4A441C4103A1AD10E2136E6A276331A067B1CB1861291FF4F54274F25E96316DEEAECB101E"
00527     "685BB21AB90CB613FD8D4C9DD69C16E8E501D19AF6D91E37C04080ED93E75F77A8BD89EB0E"
00528     "94F99CA83E1A3B63FCED36BB302E17AD569A6A74BCC7E73D2BA8E87BFE309E6E06538FF340"
00529     "3768DC9B4A6D545C0A7ABD48E72DC11B1D29247EA52144256ADA7864921071E81B2B94A90E"
00530     "5D0174F3AC5B3D093BE0E65510108383AE1C18BD74B431053541C7DB7E10512B00B94E983B"
00531     "81842AC63B11749450B34",  // PRN 27
00532     "8A9FA0D16F5292CCCCAC78FC615E47D177FEAE3B3C3593C3D9C873531142F3A8625A7D4865"
00533     "37ECDEF623D0BF39AC9BA4A8C40D601EBC7252645C1260CB1C2794F583A7A1E32E50BDE43E"
00534     "488539F08CC64B96907CD4FC4DE220279DC064EEF6491DD4EC2DBD4B1DD444F0EE6B6B5AB1"
00535     "388B3F6389283B05E2C7FE940BC72AA458CD3BCD5BA2199E193931136973EA78D1E2A6646B"
00536     "6000D4C33273F65D5E3CC04A4AFF7AEF33D05C56B24EDAC65B10A3C29B68DFB4996BBC439B"
00537     "8E0E5CA7B0F2DA8D430F00C726530B582C4EA992A6C0167F19577686125E057DF86DF78BC7"
00538     "0E2CDE25DA3AA7743B0AD608F714B1152634CBEDD1B22EBE9B4E88F6C9C1BE37CC969B5770"
00539     "DBA1AD72ED79429D5280A885D6FDDC5EA7F8BB7540D1D175173AD2468A4BDD56316D158462"
00540     "0AB85542811F9BD724608B8BA4028C82E91907DA5F0EA46620281FC105CCF3C0C676CF30C8"
00541     "A080ABE4F65B8C4CFE20F7898479B5C24929F83CC4DF3A01BE91F27C1F2C6C3A5DD2AFB398"
00542     "8B746333F358DC5B0D6E8E49876A19BBF0BCC01A1A57250F045D7C219D0481249B40F0AFAC"
00543     "38357D972B250942A3EFCFED09FF3DED7D57A9447D14A3FFF7CB69DE6B83B6034E0708677E"
00544     "9A07F7541E58D30EDA01BE99C5E6CF25528646184D49B3290D135924D251D216236EBE58DF"
00545     "07CF875052A055991DA4056D564BB96524CA4FA16A06916FA3C8FE1C6D576A8014D126FE5A"
00546     "103F76DEEDF20C4699361664208066FDE6B6B68F637B92EF08C99F6041C80FAB3C6513BE3F"
00547     "DACE09E06C498F07A4BDFD709A102C8A823121685441B7D0F68F8F42CEB1BFCF77D9ECE55B"
00548     "8E09D72565BE5D194B8BCB578EBFD17564EAB1693F25AF06F52FD24051E7C0D4B771B44017"
00549     "953AEB4B8F59BE155A008",  // PRN 28
00550     "E0BFF9020D121FB7A7B467748C46D13FB1CC76C6A6984D9D1AE83077626573D9BFBA0D52AD"
00551     "FEC3EE53A32A57617E3FE12176CEFB42685724338D31D69994AB84B6EC07C6E7F8139C7757"
00552     "3EAC93F1867F57AE8C0AE8099D89F053209B446A07C1370D0F337CA407D7C0D551773F7B04"
00553     "F2D218D309A452551FB9AA15BF929BEDDDEFFE4A8211E5EE581A2A5A7CD49937A98E95BA80"
00554     "5C5660945503A657C4CAC945F93A744E11ABDBA765BB57E439CEB8D99466B7F268DB2CA8D4"
00555     "08E952941830403BDC556395B1B469E0EBADA20E6B95DC6CA542FABA68CFD473915A8A4994"
00556     "D54355BDF8DAAE162E574ACC2D28D01D25B68EA87B8BCC741D47185E44461BC04A0479B35D"
```

```
00557     "29DB3627142B4F1CEA11BF0EB857D2644D17F4AC7008DA74EB8B7373AF6872A80E508EF712"
00558     "382A5D3C07E6C11932602D896B2A6A806DCDD0BB3E595C7F636D48A8FBD549CE12B4680EBA"
00559     "0F34CE10D81908E71D18C5217296B639E5276E713DA8C0D1323B3C0F2610F86127C465B957"
00560     "2E1F33E960B28EA2F20B7140E672578618FC77F6A5959DA75C4EFC63D3A7DF0F019E171302"
00561     "DC3E2332CA9B508EDFA9FE1B44027DC6BCB9F0F8887981A29064B503879D56720DCDFE80F2"
00562     "9D0A4A6B84FBB0712A7E30A2B0A1D707828AEAF3E74820BEBF794B00547E0F47DA978F01F8"
00563     "B7FEBF486E738A38807116B39025945D6A629639C73729CC6458CEA4432A1207EC792CBD53"
00564     "10E7481955945463BF049F568AC6C72008F3ABA9EDB11D655FAACE8479BAF73A3335C19D88"
00565     "299A7D0AB67670CC0F45CD60F7FD1EE7832037C60BAC3290B34ADDF2423CF1CB1B4442FB30"
00566     "DF081E56EBC137CDC4C58C6F9DE24104BF0439B7DAF9362FF6817B5243CD96A909953D7CE4"
00567     "94B06CFB171BDE68BD744",  // PRN 29
00568     "E0092C347E292217BB60B9003170F90C6F4DFB40F1A5BDF35F9873C768A872ADF2C8C1CFA6"
00569     "0B5F306AE4FEDCDA8FBAAAA5639452401C1E8E9DF02D5E73DFE731B98AC49956A94936FD32"
00570     "391CAF16ED9CE4D563605A9498EA7CA1E2DD9A18B928B0FD6CE0D923EB4392D0377118C160"
00571     "BD2E4DC2ECA380D40D259560B0AC23C4E6D5C658701FAAC39FFA7EECE177CC89D91FBD849F"
00572     "400F86B2AF9FFE62CDE4152E0BE8132F362A05BCF138FA4312CE98355EFB07AE50ADDE5EB4"
00573     "BC6053CB76FC12704B39719062A3DC1A236402887C80CA7328A37D4C3982215250AA505F58"
00574     "6FAEEC18B99407C5A156CBA4C04EFE4A2C8C7FA6CE48690DF2D6A5666FF81A642797EC0893"
00575     "32CD1C31ABFB25AA134B39F6262CB9ECA326E05A32C6F61771B2B1711462D4C359C4BAF872"
00576     "3643C9845AE5A4A6A8DD6EA2962482C4A7B3AD4F6605A7BBE7EC7088AF50F15BAAD86C76EA"
00577     "EC79DDB0F2EC6CD16D374F31DDCDD347E9817BF8204756754250A1C0D445C2EF4CE1ED48FA"
00578     "B988225F5365606DEB2C08FB651812459F3A48EB8A0F7C6987B0FE16D4430766D3030E8C51"
00579     "5DAE8ACB2DD3FF0035D2C27E5E0DB97BF127916C5AC1B53B6656B99BDAB4E4461BF8319705"
00580     "CD7160434BC6DADE81AE87834B030B2A21BFE91B4F13EF77FC4342372FB91A024EF01783E9"
00581     "9BE73CC30832D695022F374D41D7B758CA8C49F4116F466929BB7404279C5E42100B9470AF"
00582     "B757C1D05D877078D50FEF84CAAD49A8C5673C67438F2FF2D96978FBBD78141BF78EB96828"
00583     "6CC6BC515682682C8A8D63EBB9EAEEFFF166DC4D778F74C929E9042B48A9AB975A074BAE2C"
00584     "F8C0C6F757255E74B03DCF8176E2F603BA71E3BDD266741ACF022325800F561A2B2E442957"
00585     "35CDC018C18D75436CA34",  // PRN 30
00586     "9C6807B92D255E04EC3FFAEB477D0B1FB8589BFC29DACAC0A771C7C2770CDA8E560CD7D760"
00587     "EC19F23568C442CF5A4A99DF979C7DC0A80E046DDFD91CDA6BE48F8D5BE408FC46266ED688"
00588     "E58E57A8E467BF2BE77095AF0F6381341BE16313661A4246D5A63D82D1F5F0CB1F3051A152"
00589     "B17C14324D49E2AA02582F96B4380A652ADAAFEED1667D12E3838E0C36FE2C5B73F5DDC920"
00590     "27176CA06CDA2AF209795D7C959EF2DDF3D1418C9E4E2750F3F10CB14F43AC243A53774AC0"
00591     "11AC998A01706E304A4979C21D7966075300CDB08726130ACE32440FAED229AD95B5A0B7F4"
00592     "000318C768FDA5724459E29753FA7C359205271FD802551967FC89A303C3F5A9FA78DB000E"
00593     "CA4115BABA8D0276CCDED2BE4A5DA8D5C2A31E6EF3AF3FF3E685BA0E2745A03C4CD6397A6A"
00594     "FAA9F35D4A79C2F3FFB0CCB8E87B3557F6C2D7DEE600AFDA16C1DD5D02B26BF504540936ED"
00595     "1298D6A78E0E9C82B917371114069B176A03E3CD08C47819B08C8BB259CDE7707FF97BD47C"
00596     "17607CCF2B4DC53A9AA0E7376E89B78962A54C8EA1ADB8596853A9B35E77CF63C360315783"
00597     "6C87831EBD61B31435D8CA0DFE5F8FDD92C1D5F95DE51424D36ABF2C8F61910AE46E74116B"
00598     "8F83202B32E3496EC9E107210924EC35727A3AB6A483FE6A94435CCB0B6D5C49975A3494BF"
00599     "3AC0343910AF755F0EF81CA00FC24B6ADB09E55AD5B0C5077FB0BAF13DBCD869913DC2184D"
00600     "0D774FD4DF36A246F1145BA370BD4B93477B97A6BB0C48ACE4284CE02243449962EF25F267"
00601     "40B4873EBEDD21C94BDF8C1F01E314E511F46F6A31CC11F54B0A2B45D64ED7D32E61F524B4"
00602     "AB5D8F52732CCAB3DDDE1ED8C31F800C31CBB92F8A9365CCF985131F99F8CD1E8E37BF228A"
00603     "8D375384AAEB1A403E180",  // PRN 31
00604     "F99E9EB34B7FD437C0DCB40A389AF3D5B659FA3A0057A7101416656FB21BC92617AED9D8F9"
00605     "095DF1E562F1D10497906583E294C59B86AA92DFA1D4DC77A816C7301BFEEE49201C71CA46"
00606     "97C1642D4C7AF0453E7C3D1AA14E3F6C329A31D9830B22C2373D6EEF06B8F194C14838C6AE"
00607     "4E0D516DADBE7F26ED76E38B906D1B1428E11B1466048B83669214A7974683350AFD25405B"
00608     "4CE2225768644EF5DE9F38C602F1D2ACCFA1014F09F7C7F8DA374CC4DF72AABE0515DB7A0C"
00609     "EF22D7BD60C86B21FCB28603E4BCA9F7FC4BE57BE52DB892ED0A9465907E5AE45BCA4DEA87"
00610     "D2A6E73DCF2FAEFB2B77A163569727C898D868336C2F69A4F0635E30303000C63A3A9B0BA7"
00611     "E6ED712171FE608E34BACEB857E1D314937F9C6238830DCE856BC686BA9C3645DBB022DBCC"
00612     "8F86A4931191262CC192001FC1F5D45C23644A44F9A0BBE8EA95BE4DA4EB4EED4F1E8BF6A3"
00613     "3515F8C20487D808CBFBE142755D3014C5E1CE35EE607A1DE9E849572A6FC93FF60C516CBC"
00614     "7FCF163DF7A54CDA0720B1B43A658C2A1C42C34B50AAEC27DC9710465DB9C101BF8E2CD43A"
00615     "BD17ED52E8EAECB03F3A7FFF22F3999F9D2D35912685FA023D6C200A6FD29156AFBB79432F"
00616     "A2FF3847DF7D1F64417B910B82F52AC1DACF91E78B4395EA5AB4D3716D98B2160C3A3064FA"
00617     "49A306F81863E881A96CCD65D5816B2B64A7B2C6A09D906CDDB5BC799EE5A433496BFF9A43"
00618     "2D5429F539C74AEF39441049E651B59E45A11A751CDE8911030C9444DD4F0E1E60B9CD4488"
00619     "E405E31A641BD82D5CC8532C3BA01551F57BF4496DFBAA0BA413A2E2EC8B038875E8BFE7DD"
00620     "4E707C72D539A403A011EC5DCDC9760E3EEC518644D68E529AC88A599217CA5DAEEEE04714"
00621     "2B21F6508749BCB6DD91C",  // PRN 32
00622     "8EF641023FD4C1660E83C65765C1E7C41D8C5EB1EB85A14F1AC9939D8E8C656450D07BFAB1"
00623     "FFBA181113DA059E2D53EDADA800C04BFD23F44C144B2BA68A53C66E3DEFA76EDBBA679085"
00624     "D0DCB2479A8DB56B76B880EFBFE7942ED642FBF88701F6E425E87D5CF36A269F746FA66ECF"
00625     "1CB1583E8BAEF36AA90DAB08E9DDD296529AC25DDC6BE31E7C9C0E0C3EB8D0C9EFF6ACD7AF"
00626     "9DF6158CE75B382D2CD35F155149DCABC4B1C03935F8105CBA0C0AC84CE83B276C29F1CBF1"
00627     "319588634F92F4128F8BAA7B7D179F8485034E6997E9DD8DF476FBE851A305C59A5082B48E"
00628     "BA6BAB334F0E7F6C64626870495E13E6EBDC01C4FF544B67CA08FA319AEC58E839CEC80EDA"
00629     "2CEA9CAC7925E16000AFAD4A469D89AFC6EDDFC0498CF0D2C9BDFB927C99EEB6BB4E22FE27"
00630     "51AA72856AAA5F4C13707427FD70ED68F2817D68A632006C1C08F3C637A2ABEA9775F35991"
00631     "30D1A500B4E4AB968E6A72B8B21061758A8F80267B55F64E7D79898A71C435B9FF2CE9E43D"
00632     "AAA6FF1A335B15838E7353DF6B73482D03D3A8C33C11B2A2061B7ADB60CB2F7135D73EB3EB"
00633     "67E0A34994CE3EF985EACAFAF6725D83649012E1817CBF06885DBC4B320F8A541215AC2A99"
00634     "312C16C45882914A33718B61A0F04B125D935AC59A59E76F760E55E6043FF9A7BFA2CA9B8A"
00635     "1E077C633402AD6D9A949C5B46DE2D82E4012D0520365DC05211698CBD3E7C3426BCD2AD4A"
00636     "A1BFA1ECE0130D05C734A129A3C9A5397788B49329922F3B3860AB2C3B8C62235F4CAFD7E9"
00637     "0780C37A57073FE00D941819D6C1DD347AC32C906C2B7E5FA0A74B40FC7B797E7C041CC2B4"
00638     "4868576422D3B613BB68921CD9D78390242C211EE459556E45D0369A7B76371C2B50F1602A"
00639     "80CCC833FEB49EBC969B8",  // PRN 33
00640     "EB1EF5FED3E30E84B5745A5DEBF56482294D82C717C7DD60BA44EADA63C0F4391D0D1E1EC2"
00641     "938133F5F637B36BDC6F9AA8E9E458D56B0BEC410C1C736DD5D81C98A91A11D3BC43598E1B"
00642     "E12061C96C9E5E696D9689DE1220BE22F4B6545ED59E8DBB547F9B7A20465F3CDA2F56DFC6"
00643     "D22E049659609B99364C74501EA3D247D223EA6B7C29D8412AA92FF6E638DF61C70B81FF0D"
```

```
00644     "A0DF4EFED3F4D84689E4785D60620224F8E53C251B28CAFC8EA978E304947850D220667948"
00645     "C880E769359628710410CBB7C65FD59BFF58E7CC020E11CB72BA184C10C6CC07CAD27A6234"
00646     "BE29F5614680C7EC9CF0E39411D65B228EF6A1FBA54E12576599D74B1C7DA858F7BA89C29C"
00647     "BDFD4FACAE05B564963364C66EB88C955A345739D74944628EA55A5A0E769DF1718F0899BB"
00648     "DEA90C56EE7D54C5E297605C347A88F1B01C21233EA56E42E623C47EAEE9B1C20C0CFFB015"
00649     "988945C65D8729D831D9EABFE70E494B7462B096B07779A72F59DD003E4BCD0F81E4C4BC41"
00650     "0AD855A826EA58002DAFE56B4309E33CA6722CEA7C94E76FA814AC580342746BF446FBBD7E"
00651     "081AFFFB4825C9448D5371D25085EBCFFF0470B5EA7109E4A84BF64855FEA7EE84D65D7B80"
00652     "D327C09C1E2A22939A211B159A573B881E0B6BE5C28A89AB5C1A9B814FADEBDE7A9B9B1EEB"
00653     "2D32BF9EB27EED64C1DFB100DEC107D1C779FB7416F48E72202497CDB111D74039C58F2F1D"
00654     "39C619376DB32B9DE280EE95839A952DAA4D130BC6BAF3B61FC963F8451215194E820A69DA"
00655     "77E41F6530E9D6E801EC49990AC5D2D00A9FD24E663F8896C328F6389891C5197CA84EA373"
00656     "8499E1A99DB55EFCC974E263BA05393910F452324C7217B73DF5AB29060BCDDB7B5C1E9B6F"
00657     "207F7084BAEB56F483C24",  // PRN 34
00658     "A248DFC9DD738A7E5AC3D93E93387534AE2B707B5036BE6935C38A62E7605DE427332FAB29"
00659     "E251CD5543DB8A7938EE2967EEAB3CC7336FE0C64AEBD4E8E444A156FEBE6F5B6D97F08BFF"
00660     "1C37AE4A0F94A253A673D1638DD6010416BC0DA9E67F6E8CA6C1C0D5AAC8FA44798E72FBBD"
00661     "713DA01BB57200F17C641574E5C9A511DE9CB0E5825EE180F3B198B1D677F697AEB668C20B"
00662     "68B55580B18A14928C9841F86920A5E2625FEE4C5E16A794B354D889CC5EB8BA85419FE6A2"
00663     "B47AFBA35E1AF3D90F56176F330EDB45FA91F3328351DBEE08D79ECBCF0CC32D57CDEF9A2E"
00664     "823C58DD76AEDDBA3B73A05A3FE8011C68BB6C452BF290E3AB09788E5815F3AC0DF9BD29B6"
00665     "4110DA7E215517642290B91983C0AB0FD8923D5E80A5CCA9F79F2A0BB1A268740ABA3C8410"
00666     "F3823B2838F31F25953F41B8EDCC93A71946B56EC0B08909D101C2E1D417ECEBF06E220114"
00667     "952B2704D6D2967280393679EF76381DB071E04D86674CD340FA2A798DA90ED12D6FDEBFBF"
00668     "C92C018F6B71B09E87C991D1BA7425DCB10E988FA14C76B1F4D30C605414A40C3EBAF1C38A"
00669     "1D95A30F26EEA24900880F3932772B0333F5432039E8AF2D0974EE30A733DBC9CA1A4F62AA"
00670     "BCB23248E0EFAAA97CE93DC0A0018E3BF6972A7CEA1AC941D168AE6C071D1D4A7B532FB074"
00671     "3F9A5E2DBEBA6920B25E1C93C4828FFB4E6AFBE8ECFDA7D3D2078F46CBF3031723DD00A891"
00672     "41FA5B712CF5778704790526315F0018DCF8FAED79FD18C34F42D820745AE480F32B9E69A7"
00673     "1451960276AE39C32CECEADA41F6B5033686FDA44CA1F571E9CAD1F5BE7335C16C66134F4C"
00674     "8A20BBFBE05BB26790E85C519141757C8125CF08AD6B24AF41CCCB5D73B4DD2050DBA2F884"
00675     "E1E857CBB36858754F638",  // PRN 35
00676     "B7C32F27716B8A3100AE52B060043B6369157D6DB277069C9E90F4DE18A25FD568595AF67C"
00677     "EBEDD29E898F8445B73BA0EC2FCE412C550BE56206F9F118FFE2187A5CF703F17610A38D62"
00678     "FE1B04632A156E65462751EAB89B6F6B98CE0A3386ECDF62C5B6DFC02BDA8245A127AC9AEF"
00679     "CF7C4588CD0704A07ECCE68B34F419B03B9107D4D8CE7E6D6EE45EB3480C08A5B8FE584E6C"
00680     "CED489DB46F99107888AB5F4E49492E2EEC223BF646BC4363F45447A06328DFAD05C68EEB2"
00681     "A4F4ADFF68083C7E7DB43FE8610DF167ADAA07BF14BDFF009268A87174A96B568EE0E3B15B"
00682     "C86850EBFAB3C08D3559A41BA195289F3AFBECA053B22AE37F3DB19D7C0459E009D63D9203"
00683     "88D5218AEE6D11FE15E8CDF744807F8B1EF95E21B06338B628F32BA4174B0FA7589CF31A84"
00684     "74C9C51BD3DA8FBD92B831B9C849F4B75A0A6ADBA6C2C911C8FE5F7AED582F7D53F2BFF5D7"
00685     "063BD8A6509D2A4DE67EE3CD955DB4F2F8C268B25C4FA36ACD175BB8E718C17E4538BD82F8"
00686     "F6615FE24DB816D8C53958498613338484D85875287C141D74A698464DA0CF9AC4D00718D0"
00687     "2B4F59D12BDD149E6747D750949C7D150FB6823BB26DC33C1B90CA43218E24EA802A701909"
00688     "766B6C088CD60794A43A9035142C698FCEEA9E91FB6BC6D678EBDC6293A9285553CF8D7B51"
00689     "075DEF99A172D2CF04444AC8B16875D5969DB07B7F1EF6EEEC9AE4DDD2B18908560FB49BBF"
00690     "1151D6105FB7A547722444DC0CB1034F03A89D634C1E932D0479A1A31147D6FC4EA301682A"
00691     "888F406336AF4518714D0AB746CC79DA94E2F33FC5EE8457A16A4175B0A3FCD07B897CA0E3"
00692     "E0077131359B98BBF62CA363EA01D4C2AD5DF84CDF06EEA2F67DA1E56E46027BCE48C90753"
00693     "6474982F3F341E90333C8",  // PRN 36
00694     "8279B8CB078D12AA941FA064F72C2C727C545A653BAF5C0DB1F4BF1819AEDC4AE09A5B844A"
00695     "49692BF342D63F0D839C6283B66865E88D41858CDBB705F1552C5083D9433B98723FE914E0"
00696     "C23ACFC1A2E9121AB2CBFD47474268A7E0894C9DF8AE4DE368A4EBB7CE9A82AFDFE56F5702"
00697     "2DEB8F9CEE4A1F7EF01865C7D02C9AC3125197D073666355C05247B7CFDBCFD37C232238E1"
00698     "5DEDE0734FBD5FC95CB425FE474F05F53E0BB64BCE72A1BA583EBD70C2C45FC883B630C860"
00699     "16DC895CB5C64B134A5A93455493950FF529762068ADF5BF3086379583805AFF7145BEB9B0"
00700     "9AEF802D63A96AAE4BC148702465D89368AE54F6517C44AE557B74A6A83B332924C43363A3"
00701     "E93E2035C1C4706011042653CC7A6F41DA2EDEABAC3AE3698EC18D7E081F2FF70F325B36E5"
00702     "C3C46B8D69D2A5AC5C8E6F6F325F9917BC4CCCB03D510C4CEF3BE7E0356BB168F57CDDA940"
00703     "1DB138E388D3073D2E12B1B300972F16B2A41DEA9A3BC0841C9CD7546062525958AF820C6A"
00704     "78DA9B1712BA0C0B379A821A45CE97B8DA6ED83C877E1A6A792370E5D6396CF1BD44D4B697"
00705     "7BED5FEA84C08A44422BA1DC92BA27368B634DEDD198313A06EEE3DDD9E1C90C4955B4CDA5"
00706     "999BE46D9E6B3ECA9C0227442D5AC2D178ED9CA97E286D58FB191080318139AE4332F4EE5F"
00707     "D077561BB2FC5A35F548544340FBE617601BB7F35855EDB147382BD22D6B8FC84995BF60D0"
00708     "5D98D7D109F87A4DFEC1379307986508BB90DBDF29FBB19DA552C8F41C28871DD4007576B0"
00709     "157E5BFA13BC1C7F694763C3DB61FD02032D31A3BE7162341C7D0D4A96AE56125865090560"
00710     "D9F82611EFF34A0835A1365E3FAB2B802FDF358681D7FFF1B7E1A38670F1535AF0D1831CD4"
00711     "CA1ACA1CEEE8D8B158B30",  // PRN 37
00712     "DCF8B1B7E6745224196427D5AD4AEBF6C6447F9D38A4BF29C12DBC07DC2185232B1D0BD7189"
00713     "8F7B08AD8DA2517524A597EFAD7D331BE4661968E1F8C43FD2A094B42BE350128FC4C1C4A4"
00714     "28CBF888AE39E129463A2177FC1DEF2AFA62C95F909AE17DCF4032329F38AAE29AB13FEBD3"
00715     "8FD3DA208CB395BD476F33BD8C2FCAB7F1D2450B002A6154E110AA2FBBB2C379531690C111"
00716     "3C51CB5D4175FA6925530C0E8853606928670BFF8DDF90154F60D2615E7E87D6224C19015"
00717     "3FEEE0434C6949D82EDF633780F82836993E4D839B400372EAE98EBE21349AD245728977A6"
00718     "C78B358467769AF9BEF0168A7126B000331D28ED1316AF7EE798278EBCAD7847A7D72C6867"
00719     "9DB4813CF67A7AFBBA3CDDA285A9E284D27D263D22345DB22D0334AAA630FF88597474AC86"
00720     "7B0DFD43A47C461270EFEA09C971C000EC353ED5A408C9F442CE32FB0181A18A5D5D72A5BF"
00721     "621FEC23BDAEB60649BE3309B8A208A61B29512D823C5A066E4B2DFCC6F53FA2FE4BBE14B6"
00722     "3A9E0368A7E23FEA409A930A1FA3ED41DCE2AF953EE06E7BC5EB3EF4E479CC144F6F8EA123"
00723     "8EC7B0EE0216F6CE75110BDC4B8224B08466C5FD8B0FBD7CB52DD9F92A3ACF9AED1F265A69"
00724     "0683374EDEDB1BDB8CE9F6FBB9C8B12D7C2E4ABA5C48EE6B1491BE66BD1A485CBF07EEB932"
00725     "A0CC24482E38DD58A9A20CE3B225B4D1E66695418FF20ED2F6098523CECE06B5615DB4ADA0"
00726     "DF999D717A3229895A9EC0288035D2CB5307D5B9B27652DD2EBDECE8531074B828C6BE1524"
00727     "2B2C41674CF5BEF9B1B2E5E05DD93F398737DD1E6BAD21A6552C7C5C376A92A89EBD5A11F4"
00728     "ACE4DCA3925DB5DBDFB1CC0FF7D1207D0F5C0579B39A32899A05C4A7C5E6240434D017441C"
00729     "B703101DE3862122C9E44",  // PRN 38
00730     "DD5F5D29A86DDCBA09CC373F11C14747D12C01974E74CC7C06BDDCD5F2D45256FD6964D932"
```

```
00731     "87B8DCBE414D74C6EF98F1CD72111DC32815E03E13B2B6F85785E5871264BEAD6F8BAAFFC9"
00732     "7E7650EC187BECBFF59925258A84A671791C33619FE983EC5868FE5C377D91C690BEFAC9F1"
00733     "9AE1FD9B5C4E2E0D0C52A2AF3F631C5A80BFDDC16EAE7ED73F66A69791E8DD2815848D6BDB"
00734     "4E747D6DC08FA6B606876C3D1A7AC9E344DE02744081E0C0CB522B091439F6263D40E7DBAA"
00735     "159D31BF49BAAD94BB974C273E037E49EF6826C8E1142B768A2E606F3A76204523024B0D6B"
00736     "E9B3A19032A5B877590E2EDFEBD880BAE7E22E439855D2B0217974A6D7C6F80C3B0D232ED4"
00737     "96F931E5E84A3FF6BAD1AB581EE9723DD70C0A794C25281CC55207EE31D34C91A1A4B74D73"
00738     "6A0859E62B25A93DB39998D26A1EAB3E4143F01B664858183F23238A26EE2047FD1FBF526C"
00739     "D6E3CB6017820E8F83495B4DD93B8D70DFF1C06AFCAF11E81E904B37F717A98DE354D45465"
00740     "741BA1B4F8B72A612582A2A1F2E40A05727E147238152128F03FE0F13AD4B3CDB2699CCCCF"
00741     "D2FFC2D6509C01FD4B8E7AF42D3F2DC2F7F0FC6230836BC2661F11CE7838D636D7A931C62E"
00742     "891F23A37FC7642BC94438BEB34C28AE6F000C1C334C41D5C0E29F9DA1C0FCB5F8FAF77F78"
00743     "684170A26962015E86761F17AE5D621B35C8D92EAA6BAE8C1A30B356386B99693A725BABC3"
00744     "574E265E4B0819502B886620D4D10DE43ECD300CBBB5CEE96F5CAE0567BFA11E4AC8CC1288"
00745     "A1A90E6B191CB128402C6B09A8189917AE49D14AF54F7AA9314F0423649DC7A575E731C809"
00746     "9ADE2AB3EEAE1F8F4CC677F920B664AF49155824C08A544DA3179C436D52536FC0B186B521"
00747     "2FB36AB29765CCA6BFD5C",  // PRN 39
00748     "87F0F71E090A523A05795BB3946AAE7225D4123530FE4930BA0570C51495C6E8C674AD68A0"
00749     "B1F9DF470C752F897677332299D7875F4969820512161FCD3E6183AB96E296DD58DD06F3D9"
00750     "FC28D934545E5DE6E667E41D23383F68FAEFB13139D1822AB1A50708E4598E0B877D08E004"
00751     "A75C75268FCA08BAC51532F3332CCE45DE81F112CD07ED277D1407AABFFA852A21BC1B11FC"
00752     "F906C8A24CE101E048606279EF6CFDFF73C6E69CF6C26C4FCC3DECE232FA9900B5D16AF9E7"
00753     "6F54F35718583F746CE3ACD0D9DCDC3495C63CFABB2D8FD15845D7FA1BE3E1E04B403F5572"
00754     "F745EE74B0E3D6E912EC7C9FA465B14FD1CDDCAC6D35DCAF09ABCBB78B7752B17382046D94"
00755     "EAD4EEFC56FA0B92DE2B690801C16117B8A1772D9634E13AB0314FFF209DBDAA17BF512FE6"
00756     "BBB745641BF8799C1EA745264A2E9AF3F5EA28F01449E2D656785138 06A3CD198CAC919971"
00757     "2FB577DE5376BE375018547D4D143B4C7D87705604DF1335EF12029CB54D06692A8EEF5D51"
00758     "9A96F6A448B32D663EABBC6EE600244137847F367C12C245E06F3D963D00CEE03A308EDA0A"
00759     "D072E45F787F4DF945B011D76801A5599FDE954E257CFB993E2E15A8323370E6424A82668E"
00760     "4EB080708FB239416EAE3D235355198498A8CA3C630A864503716DE942937B462B80CD2F6C"
00761     "6E0294CD708E85F39D25377FF63C43147629876C6192EA1C34ACE520D7FB71AD03716641CF"
00762     "901206491096125B0E33252E7172BE6EDFB583E6034F3A72FD01D9C8D091F78BE4975577A6"
00763     "006128AEAE8DBD7E5AD97A7DEF7F519422802E2E9A5CB80C7E871CAD8579A6691B44660784"
00764     "8D48E9583FF341E9B6CD5D4A128CCDF534E9666E4EBB929EECDCC03408A5F95276B65C0996"
00765     "2C2A1DC2585B617C527F0",  // PRN 40
00766     "82A6C8E99391CCEDA8359F2DC980A844B26BEC60C7312CB540C3196433C05DDBBD2B6D43CD"
00767     "89D91195B9AE50B5382269A50B63EC8D3C32354EDBF22C4988CE66E76CC63D97C9EEF15C1F"
00768     "0714035ACC70F809BB82AC0E33C47F672BD54B133E160EC5C0B3857A69FFCB00968650455D"
00769     "7D99A689C9071929DDBC1C552623187D6F5A7F1E7E7A4CAE0571DBEE0CA1120184E545DCE5"
00770     "42162227E485344FC12FB10850DF040B5DE5B5D7FF45A414E1E4BC61AE8BE695CB805B0FB8"
00771     "48D39B12A61428A53D5F610DF4D62101D22E4BDFD1ACA5B06B6A6725364812F963C81300A1"
00772     "9414AC98C7043C3D96FED8577A069FCBB2CACCA0CA2B28537834FAB53887CAD2C9BC10761A"
00773     "132C12893012F48671FE28702497BC2A39A3A5BC0E866A73E53FE2FB90F1749A3050B9554E"
00774     "D499BCF279DC427CE6499FADD5AA64C5964E4C676F9749927145244E8CC1A71BCCF4BB6236"
00775     "6BC9CD2DB76D2E3FB4A56AD4E74CDC48247A3014CFD376B2E650325CBD7C6ED016A4941EAF"
00776     "71FDB91C2809E9E01A728C9F48851BBC4EADABB23FB3094CF69D08146B2305116D68A591DF"
00777     "11362E22D537323D0BF90DDA00B6BAB7B0A200FFD7BD0AE2B12B9A99D581CBF43F9EB06A8B"
00778     "F59885A36CB112DA1F78D2513B313951088A6AFA2255B6FA69B40CD9C8BFE0BDB837CF9550"
00779     "44FA561FB3030FA6E04FCDFE2B3DAADA5BC4F77F71C780352CC7AA7D051F663A13C45C6443"
00780     "3832336F0A9AC78A62C00777754489B6253EFA75EC6DEAF2A9EAF89FCDFA28AA9D43E39E25"
00781     "FC0FD3DB942F756C5D3BB437EF7C9ABF30FB4DA227BFF021E1CF783FF1B8BC7BE2FF6553FC"
00782     "84FB9A93E15CB8E07D503D1CFD00B4D8154BCB075EC100CC635B6615DF5FA558F4D58E0EA9"
00783     "8200B97C9CEB1C80CA850",  // PRN 41
00784     "EAE3C7B357368A15F87B0872EB86194169108BB59B4BFC6C8BE058C57FCE77C99502266EDD"
00785     "2DF7335DC5BB3DACB0C0A042BF0F46B3773487F811B27103B791F7099F58F43FC82E682AA9"
00786     "09EDA6A9F8BE4669D3185A704FDB3FC9C3E312612C3A2DF427C120AE3BB5FED4634E98E7EC"
00787     "9EAC666E317773FDA171B52E139336C70675CD65FEFB6812726C075FC9D1F822351B4E2234"
00788     "A754ED28B3FB3541AC0C26306AEC93EDECEF769AD8BD106A484FAC54D0AE027711D7E50096"
00789     "CE24D2DD2015C50611AF4EADFF16BD8BCFC7B9989EC1235FB1F7E6A72AF7A8AFCBF2321306"
00790     "702D3DAE840127F31926D71764B771CA482C7ADCB4578C5A0919DC5E7F042CE2549B3EA3E7"
00791     "D68FA12716695F765AC2950BA62602F579412D515C24D7DCA9E66B6AF120B94863C9B701AA"
00792     "D75EB3AF04BD7D32F330946AADB23F5059 53F2B5370CDD277736CA17E0FD4457934A2CE132"
00793     "FADB6A603D0EB3706629B1EB1407F794C2018B0C70074BE4061C909D5D760AD42FDC10089C"
00794     "D94F94F7B4CADB2263636BC2964884702DFD17A750B5C1129A120312EDDE0FA430E2AD5134"
00795     "5ACC5F6DF2467EDA1875868DE3FFC9988F1536D0124ECC339EA2FB3C6A4D8A81710CBD6263"
00796     "C6617928A21729B2B983BEF6E0904CE678902E41C2DE8810F2DD02021BABB046D4559566E1"
00797     "82B0299745 73F5AA03E061D3A93220D626FB6A3C3EEB4F4C92680CBE244071F4A800958F0F"
00798     "30A01B02D0D1E78ED5FD88EF518FFC9903EF838133CC4A5687BFA9B3F4D74DA8B0BB74ADA0"
00799     "5C276E66BA76EFF8D255F667C4F007F70A7627AB6A9A43F18472FBAE698681E9F2E2B99B8F"
00800     "500D6BB548F5C7E5A3E915A9A927AFB240B47DEE337573161193890 5202CD19A4BB184B068"
00801     "93C5A1EC3C2DA490F0AC4",  // PRN 42
00802     "81D7FDADDA0B9157C00450C3DA7DAE5E0CBCBCEE93039EF567E81961E115067C80BE548326"
00803     "1DCF0FDE08A5C1EA39CDE8B88C04F34225B7E4AB0A2DC241A72B9A2717A7525320C08ED62C"
00804     "D6E9B8C1A4E287B12CBE6503376C9EFC9B2267663E7DEB56E55B54D2E91623EFDC32C81D85"
00805     "3DE51E73260B3FE1AF8DE75D8A9D0D418496C3F8D7A73317964ADFE9A0D45A5EFFDB81D7D7"
00806     "D5DBF5E2CDAF815D4B7DE87A2C28FA6D8311F0909D6C6FE2D06AEA7F98E4DA3581F0F2F47F"
00807     "328608E6D84EA2F9F2A5A4FF121BF8D1522C90D2D5F3CEF9E3781EB5D4E385E9A8C58530A7"
00808     "9FE70E1399EFD0BA4FE93758BA67422B037AB39F4DD20FA848D5B255511F2F820C2B5B1187"
00809     "57B09DDD262E100D295E5B74113D5DA75FFB7049E5E5D32C5C3CDA6B2BA503F48C82B95A76"
00810     "D9184ED3E915722261D6A1F004DA84DD274B556B17CD1FFB158B22C393CC06A7D6720ABEE0"
00811     "861021B98247F4A06FC04A28CCCEFEB7FBCE802E083410A646B29A37B732B99E2F90C1CEBE"
00812     "BA828B2AC07E8AA9C32038D8C7C7CD824ECEBC834DAEC4C80D303329D5F5389ECB4599380E"
00813     "A68EE5D5601EB212D4ED327B75C407F3A0C0E7B57820419F377288D4F04F4E53458931A51B"
00814     "EAA87CD3725E6958F03CA9E6C7B8598E445F4CE28379A76EA006DF773ECA22C9597501A5B1"
00815     "CB509C040D9375A966B7BC35CF5C575A4449F1A307E4CE64BF01566A7D4CED5BA410687B43"
00816     "E3D35521D03054C37B1E7D14E664C5442E1100F8DB7477270E39FB30BE495FE254CFF3E2C3"
00817     "FF133848BD75846EAA412779AD1C8455A7A5F1C951230826DD80165C4EB4BA8826E09BAB44"
```

```
00818     "6CAB69493974D7A88C3152A4D25E1540C716E669A5B0D70B1EE58836D8C0F431981F019B6B"
00819     "6724F4B7FD17086D81E48",  // PRN 43
00820     "B67814C98514A451ED8FF4C8FA70190D6087809301EB8D5D17BA587BDB954D631907779697"
00821     "28C625B111A216C561E796BE9886EDEB000C727709151169EA8EC67CE997F748DB608E2D61"
00822     "3140A787B579E31A5F9E92994A3C9DC21CBA91AA6B6742E0332A9C538D366DF211A7219864"
00823     "53439CF1BA8D49D940A2A04293927B1094CC9EE1E33CBCE7678A676E33F7CD5DE8D93D1C0D"
00824     "5AF9BB9D6F7A51FF9EC56B80D83208FE71B0C85F3BFA60ED013BA0191E07D2B15AEF426A71"
00825     "22D3E754AE749E17E6DB43C772A71D7ED7645FF24760F83F2AD6448A7A14C2A2472032BA36"
00826     "B3238C5AABEADEF6240CBE68606954B60051D393A3BB014B0D5386097A02399E5BE3866670"
00827     "86B1541D939EF67620B6D83EB004C6C71FBC6EB3DA8EC6E50A6B05D7BD4CDA52CF441AE032"
00828     "E693C3C297E0C350CFE95951DAF2E1E56B04C232B4765A2D0B4E5264BA5B8B4A3563EFEC09"
00829     "8B9F01A7ABB5C6F146B828E47BACBE4117CACDF2B2C882C01F787F52A30476881EF03A47DD"
00830     "63F98804CD8BF8A77549F4CF893B1E5CEE4F2A7843AD78145AF3F186DD7F0EDFCB5AD04D59"
00831     "8095A357CB7FCF10FEF6E9B891D1AF98803A5DCB68AEC494B684054C13C8EE429458A0E90A"
00832     "126EC823C02B7E7509F5F344C679B670B69E7D290A7853EBDF9304A66E7AFA4A16FA412F3F"
00833     "F810BBA9A75DA1A3E3EF8BBAEA081B76D5C935132497112E36501554B0177957D5FF886581"
00834     "7154C11DEF9F42ED0401BE65E060183777796B29E163DB4B800A874052E2839AC97970CD38"
00835     "ADF8834C56FA78DE936A366B931DDE81992AC1A8ED5DD54CE7FCD8F7578CC725DDE4C8ADCC"
00836     "DB3DAED876946D166107F088C5B48AF08FCFD419A4A0E0B9DBBC3A4A3BD4C5719658BF28BD"
00837     "B6D036E82D02EF9822208",  // PRN 44
00838     "EABABFF30EFC6AE2E63A7F47BBC4728B986F66463D3781C97BCFA1C6308F1BF7A47860A41B"
00839     "0601E35C49CC72C7C1069DABF2D0FA2DE88F83FEFA369C447AC31483890EAAA9B3FE321CAD"
00840     "3421D8DB9AC305102A9393848098482BA907E816BA6BFBC386D99191DF8E4DE74919DAD8B2"
00841     "7A303D4F344995254E036C3B7FDEDBF1719BB6D48A385B055867EF3D9B4C10D9F1A08558AF"
00842     "D7ADD124C31AE9387EF72B3AE22CCD91B8964DBDE71E142D399B9E0FEEBB86A219D6E04278"
00843     "6A1573F495E5469401961E9AF48CAC31C55561B75BE83245F7CA56EDE68AD2A9CB3C04F920"
00844     "6526D886E2E45AB8806728309D13C23A2F8D15D291DCF7F0A84D11301999E0C545A06A5628"
00845     "47BF179FC3FB40401C8D20D479BA89A595C6CCF591E0828A457BB3DC0EDE6648B516788582"
00846     "E55C22D8F8A84CE4CDE3C749CFF7F8F8540CA8A2A1B472FE149313DC5F4EF7647D5228CF13"
00847     "CC8414B6FBB5F5AF69173D30E4CF90310A7F2070E0A6AE8A425FA381B8333EEDC45B3B887A"
00848     "9AD687E3FD5A9413A68BDC144193D9AB0F1A10299AC0290C7532A5B61BB8EFA1E48F188AC8"
00849     "83072AFDA5A4E0D494E7CCF900FC751151691A06D29486F2CB984AD078910C6D39BFF75C5F"
00850     "481C15733FA0EFED29278BE0517AA25240F8C736A04F11FF68A1077C36E54B0CA2617211DB"
00851     "B5A819290C2624D6CB778767654FF61C4B820CDA94F4C7F59F612D6BE3BFD1119560C8715F"
00852     "A50B0A5EB921AE8B9E15BE5F6903D7B1B607575DA92CA93B3388916D9843351F33E97B5EBA"
00853     "DC4B30FFC8852DD6EAB0F63352EA7573335EBDCC31FAB5CD988F3F8D62B5E32B8EDA81BAB6"
00854     "5646158BA57FA37237F575C101450C223A734CDC13262C4FD4E8914519FAEA599F5DD3A17C"
00855     "44727A8DE22320B2F91F4",  // PRN 45
00856     "998031EE03AF13E73A983242CF8102737EAA04F5C475FE94EDEE67570C1794910266F096F2"
00857     "4B02A1C9DA33B7C07442A49FBD6BDCB58556655529CB1B4056C159AD1D8D2C0E9861DDF78A"
00858     "DFF68A8FC35A1BC798C1246C5B15B1E9820F1F960D24017250B0781480127733FF1B67FC30"
00859     "A6CEBA8C66A8DBF09ED2225AC25A790F1080791A93D73CE1793A0E4070E326E8071F1E5C3C"
00860     "A70FC475248DD1A4F623413E1DF6154128B9E9CEE4361FCC269C32A54BDFB1F2AD1E7BBEC7"
00861     "0DB73CF27014825BB5A18BF924D1DBDD6C7C929DBCE3E46DA1FA260245BAD192A7B56A703E"
00862     "C22E51DC2F58D00EB249638571E2EEE1EF396C04E740D9F5D44AABB160EF2DAFA01B725A6B"
00863     "45478FB55CBBC24AF4E4AFFCB5671658DB913B43B134C68A291A4C876CBC811BC05D68B2C0"
00864     "73DD510C26E490B9EAD5F050401D16BA7405E570D7F2DD874865492603959B578153AAB478"
00865     "AC4FE885E1136B62C947BB0C867C6E8B87A4D9C38C143A13C7E3B954DC1B66BF3714512F46"
00866     "B6E4EF9DBE4CFDF5A7927B33A9E56CCBD56591593905BCC7F4419901281B14F01D4CF56337"
00867     "FA90613893F12E7BA7731ED016C1FBDA14BB33E5DFB9DD2B1761733CF417320B1D568C32B7"
00868     "F7BA02EB59183AEBF5218C6BA3895133E7E9E80F8D25BB0B2AE72F0294CFA3181388633642"
00869     "E487BFD4631F2C81F2593D10127714D2493869E81AC9432AF5FE5F7F1B2C2AD813A5502823"
00870     "8D2D6AD9A26D9FED2064214F294E0AF3514535F56F7F659B814B7ACC8D7D13EEEE5777C971"
00871     "D94C657F88888DD478A3565530F7DDD4232D21A4A25D5D36214D68417FBF893ECD442FF657"
00872     "8B0F702F1BC07349E873F4F552F48926BAC6282FCF3998161035762F38C93C4108C7BA43D8"
00873     "5F76A78720A5D97865128",  // PRN 46
00874     "D58F19B469D3B8182F9EBABAB5E2B3A2E7BB74601BAA44FBB77AADD78C9366DDEA04F21AD2"
00875     "8FB0E71DEE8A169D41116E63F56DD3ED0E07FA94DDA8ADED454980BB900E8DF6FFE21409B5"
00876     "82F5AFD684B5DEF4A875C3ECCFD18C4B27A7A80F3FD6480BD09C738E0040C234FA1F5881C9"
00877     "70E1670000169345C587D2A7817FEF71C1CF9F173924F39F3D76107A526358FC2520A59DC75"
00878     "3DA8AB9FE705F339888BD797B433A5070E8E33BA0C9259614B7B89D69F4F541A5BF9B2BCF4"
00879     "F58431FE2B0D6809804C005C43D3308A97308B3B9DF6FF1150AB170AA54261C088C2A6C6CB"
00880     "E26D3B7B6703E3A565B4DBA21A09813F0C19EC8F801A34ED9F91A6114D465BB7E9E04FC15A"
00881     "369654D15D78E125BCF43E2A16FDD875366806448F9B10C6F176F39A9B4ED27D2AF8FDCCA6"
00882     "8DA95E96B2342902D9FA7A07F1D2C2EAEC7B91FF3BAF211FC69D8C72AEEF5BDC767E7D1DD3"
00883     "1390110936BE0535EAA37B3B2E11EFEF3FDBDA3D90A9F264508F7B2A172D95C3C43EFBAE0A"
00884     "6E706361BE044A157352A719B25A26676C9161AAE58E8A452BA0561AA27DD5486702FE2031"
00885     "A55668CF2D92A9200D08192EAE64691268C3D50B5248A3F1714BCB3825A016F6888C2998C2"
00886     "9F8129CA6BC2D5D5B3A9CE8425FCD39C84E5F311ABF8DFC84EF58543B3FDE633124B329F04"
00887     "2AA398D9386631A41464E6B32527435C342004A8FD78C175031E43736F060DCF43DEC4E594"
00888     "F18599121BE6E817B518A93C803EAA1AA807530EB7E27492EEC49470FE4AC65833EB4EA3EE"
00889     "8CB64A13D1AB688FB81609F068A876669D618CA663638715B2F0ADB5D4254D797F19D87DBA"
00890     "270DEE72ABCD5C402C174E8F88C8D99EE327952D0CEF2B69C2D18DAAE031A42E2C7790F59A"
00891     "77C58879BF16A720FA334",  // PRN 47
00892     "E55B50808920F178FA3DDF8259E9DC69921D014969A7256D744B6B55D5D9C90027C2D748AE"
00893     "72962A945338C153701F3803050BE7791704A0FE0C3A2021F2B3381694C64A9344EFC9ECA9"
00894     "8DE06233AEED3C0E5C210866B0CF6804D356122AD5B7F3348FB764850488DFA8E37F70D86D"
00895     "C87DE5150F408C6E8E6BE0DFC2DCB1E77F9B60548B4657A0A8B7FAB7736480FE88A72B89CF"
00896     "6EFA7B18DBF43E611CA7B61145179FC6293DA3B5373C82DE82EB80ED22775764259B8BF6CC"
00897     "5928B92A038C28D73B7C146EB4B5C2F4DB9159ED1C5F3C99D73EC91EA22DD78844AA60E8F1"
00898     "22E6EC73D241CEAF9EB48F5937351B6BB074A43F727BB05578F37C3E962769AFAD89E237BD"
00899     "B06C74534AF5B3D5CCCE7B73040D589E9F9A0EC0D58A51706425A02B2637620ED29C5FB1A7"
00900     "A12B1BD79A0C42C60BC9FB46200C19619F57A7AE12248FED59BD529CBBAE3DBDC8DF43D8F8"
00901     "42CA54824B5712C0698523C6F2FDD38F93026DE1E474DF762A41EF58245A70A91B6B22A6BF"
00902     "4EE6CF3DF6384FC2F6CC4A462D025864BEE4BA550DB1BD956E85B18954CC86075118A384BE"
00903     "EFDE6063CCF4BCACC57A890494A8251863FB1E464FD711DCB1C0A5CBE9456198AF4B0CD942"
00904     "76EEB6C96948B939439D2580F5E1F1A58304B515DF47EF808BCEBDA532C67C18D5BBF7CA07"
```

```
00905     "1C7614681285889DDEEC096AF7C133083CA787D7F3D9A470BC81B151F3FA195680F0679EC6"
00906     "88ACED82D41E3EAE622B5F3E11A3B892687722CBEA99A0A28C8FEC5E9FCD0723EB81A43F6F"
00907     "66047FD136EB294E705F25A85C95AD8470273DC3B3CFD2294B69099154236EE9588EBA15AD"
00908     "C0BA2059040C9C194821BC290FFDADF97F6F5DB9FAA9598A8FB378F3C442009D27A30FCBCE"
00909     "A4B19D19BCC17000EDEE4",  // PRN 48
00910     "C62BFBD744BD9DADBE977CFBD96E58F9545D823DB6888B0C6345FA6C21D5AC83FF9BEE1107"
00911     "2679F000415177 57C476A876AA62280F8976DEA8201A7372719C54511143DC6344EC05603D"
00912     "3E70CF0D5252D53826D84444F7C89B12A5BC439248BF7ABA05CE5575881292428C83A59787"
00913     "F81BCD2C95270A13AEAA518EDEBFDFE798B28FDA50AEEC08973FE3AC395074A8D33ED0B781"
00914     "975BBEE0DB7CB6D7EA6BECE34BAD17EB538A636C2D174E031DB6D26B4C06330107859 53BD7"
00915     "73266430B12B14D1FA3CA8D0FFE23CD135B9368A86C01ADB25D9A98317F1A1271B7EBACAB6"
00916     "4097DD406798157623BE40DF8CF1F888E117CE8476C6A878EC5083B9821EF0A65FA51DBAFD"
00917     "ACC3476CCB2FBF1F5B6012BC539CFAB06C5B90C7D5C3AE701A25CE859B0AEA2BD3150B530F"
00918     "AD35C506C3249152EABAD7772715528640A6DFA24C19C945A51C4D76B7114116A1D82FC9E0"
00919     "6E1C72722CD2C92F11D78645737CB634C383148C3B085B6075BCEAD9E68E4636E0383C0A2D"
00920     "AEF63CBC8E1E93A70EBB8471D95D2663DA8BA5B4D061FCE0F144FD1BD9D94549986819ABB2"
00921     "FCA20AE20D6BA2D2F11F0C1CE8D2B35A9A5341E509B9276F59271907063D8056B1EF5793F2"
00922     "136C32D113F6B1860C9FCB6C6AAEBB7F0FF95519A0D6977F8DAFE137700C19B8006A7A3F65"
00923     "3BD6DA45380A8A604D317CEBA9C2C265A9DBD6D18E9F8441BC03E4F9BFE04A9DB0CD818ADE"
00924     "8D831731A9CECF11F440886F09D3863062F4D92BE0142886765F2661E3051C5A67BD3502B2"
00925     "8F362130E4FB6F0FBF76BADD04A4563187D926E36BC61D88AA5E68A1A127C6B21B6FDF436A"
00926     "CC6AB56BE5EDBC217287280696169BAF2810EEAFE785AD047DE2A6BF2C060F6FC044F5C337"
00927     "716953273BE8DE7FE3184",  // PRN 49
00928     "FE62E6E5534E3DB8EC780270A7C33FADC4A612A91B62293B2B89A78CE2196AF2D0C693EA70"
00929     "A2D61A81D295E87FE8A930659AF517CE6F311EE423B2F1076560CA0B4602B060114FA00C6E"
00930     "681DF6A9DF7DEEEB9314E1B022BD714D90BABF6A2E1262A6F01BA663CAA0CEFF977A15DD3E"
00931     "564A8EAB39F72F60EFAD18DEAF2B1702441F262155B3E63A19A9715A76B276A24347C0BDE7"
00932     "2D74D0EDC29B8C1D8EE902AD3B5295DB074AAEF0995F30BAC3FE4BA8C737E5BC2B9D1FABA5"
00933     "E054354E47FAB7F960F3A939FAE76B4D525971D18A5DB7F50ECE91B9AB8364E5B6A3EF874B"
00934     "9616237724B8A473C7D50339D0BBA17D241AEA85DA941C52F79DCAFE52DE5A618A3ACBC2FD"
00935     "CE17C0E603F7DC10B59E05235C802D96B36C924FF54C7C0418A0E76EE77E70D75C1995C0DE"
00936     "224CB65833067784E872AC3340A2950DAD26F1FC373F71DA4D1DF6069C1C7F5F772393BFF6"
00937     "F3524530E69857E60AFFFD213DCA2614653C2064DF76234C8B5FA1303CB0CA9C35BD9EF5D8"
00938     "8E3347720822B3C06DE44509871339133132602D4228AA960800A2F5883CD40327F9C1F823"
00939     "BA56ED8ECBFDF7CFE0D26A21BFE9E1C6A86298840B6E233CDA4DF652B7F0485946AF349CA0"
00940     "C6738DAB1DF32559D7F623D7CF74F48809B9731A918C21B1F3242F8A4812F1E558F8DEA7E6"
00941     "8162ED00096163C8FF6317D6A19326F9606374453F61B7E7816F7DAB45A04C8C687A218425"
00942     "CACA9D94A185BDA4B6C9B7E30EBEAE6E992A6F5590587B06C2B73024E09C46104750B4B446"
00943     "A971B9568C4C29C036401CCC36A6580DDE2E0ACC4509D6D0D4E1E7E6012B01411487673DDE"
00944     "5A134BB226BBE1E508F0A7CE3A60BD85CD528CDC8D2D86E27A27477C83F6E1A9CAC0AB73AB"
00945     "D25D2CA84D26BAFE257E4"  // PRN 50
00946 };
00947
00948
00949 constexpr size_t GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH = 1279;
00950 constexpr char GALILEO_E6_C_PRIMARY_CODE[GALILEO_E6_NUMBER_OF_CODES][1280] = {
00951     "F5A3D656F9DAC534C05240934AEDC5F175B4D4873FA63FDBB4DC03D20321BE8E78FD2B6525"
00952     "51FE532214B660B493575402DC6F93F68B3A22B5099899A21DFC979C3A955A68B6C9EAD355"
00953     "009D6C30036FC1C0DA770D6C598CBC95CC2710DD627B5E10CF9764825427FE0387DF7EC6F8"
00954     "2D04F3BA9FB420C3EE731BBB5E39D5B9A52D6E74E1B359D94F31A1A6583CDDE1D34CD5B18D"
00955     "2F66E21E6E969BA981631F0C9BE31C8BBB66BBBE411EE227BFE9142C851A4EFD9D8B4C42A2"
00956     "C2480B47B39FA395D29C329F0016CE57943AFC6E927B4391441C4E2C0475AE9C9D765AED39"
00957     "4DB059623D8C1288BE6DF72C69C6AED2AAA5F8BD4A978C4A3AB6D64FF8B0FC1AD4956E183C"
00958     "047041310390C0F39F6AC03E20DF72A88121182170310B7B72E742743FC49DFBC34FDFBC38"
00959     "0375457DC4147E78BD655EB3BF5D98A4BF788A8E9AA8DA792359F2E01F6E368886D294958C"
00960     "8A9EF481BB6C7AB4DA9D45348815609000 66DE707B81F873D6607DAA5EAC4C4EA9D1B1B81E"
00961     "83A69E7ECE07A3B525EFC8FC1D16B42A3BA6B215E87C1698D15E14A066045C8CE1E49E39A6"
00962     "F54F473F6ED2D8A9E777FC39B2321E71BA999B11BB17D5C62894C5125A687EAA0A4A30817E"
00963     "C1BF7D7A5EC983029218255D4558A6784619444B916238CDF511179464272309E5F1BE2AE9"
00964     "4FE7FB2578415A1AAF7414EA7100F9D7E9C886D148D4DB3E40B8761857F30CBE06E0952EEB"
00965     "D9425671A34381EEF7CD9F43B35E3E8563885332CB1F0DD358EBD237D513E0EA6D878001AB"
00966     "CD3138270F9A84B6D223FAF18B959B1E725F85D4DA26A127217F1791E8CEA41D307F38ACC7"
00967     "16068A1D0CB76F8337D32B9602297C0E23BA115679B219A4D5936F5A1E7A03F2AAA77464AF"
00968     "A89B278C63865F672784C",  // PRN 01
00969     "FAF0C60E496650198BFC9A6439703908202698930C129C4F44DFF72FB0BB4CD682B432B372"
00970     "59394EB345B0170CAA2B80CF697DD6A748F16A57443AD90414D29E2F08585261256D601F25"
00971     "C69A1DDB6E1E2A84CFEC805E231ABB2636806CBE410E91F77D39BE7D2E2561C14AA6255F70"
00972     "893888BFA75E678B64CF24A6E58D6995BCE88D52EA67D235D142A7087624D70487B09D5DEC"
00973     "BE0394BFAB0AE4BA70C9FE89BFF26FCCD8D0967DB8F8DEE946E351D682EDE22858DC551A0D"
00974     "7DEA29488C68A8DC47681284A1EA38365D7DE5040453208BF0427722B8FBEF5C945DA0053F"
00975     "6E86AAEF688E36E839480D88119B98D724791825087E05A460DEF21C8C5606A12CA60DF8FE"
00976     "6CCED0894CC82200350A4EFD1CE37D9E0F638546CA829E1BB73B59B324CDF74395FC2AD279"
00977     "8CB6F4A3E672AF65F107BAC8F03600A6705F13C9E61E249859D1A80AF1C1DC65586FD5DE0A"
00978     "F497DD14530071EDEE0EC495C29160B62B536613CDE573AAEF518086C1FF759F6D140633C7"
00979     "4431305DAB5C78ED02A62780BFF8D0DD8DBD3A8ABBA7476E0233EF37C483AF79BAA7467014"
00980     "9E718B032E782F6A0A0A344CE0E9A74C6707218ABF4732912E1FAD286D8451DC6CB4BAB5DA"
00981     "F13F6BC88D784C2F768E77679C97EBAEDBB0FADC2FAFC767B6BF8EE2E75530DAB81752A318"
00982     "D1A20409A6701C7DECAC13E5403EF6B6BC7F08E0E71834AB6AF60F10756F75B12020681F3C"
00983     "5FED95DEC068CB94DDB76EE356A9755E68C98ABF565A043FB0D5D78DA229EBAB7B8A09334B"
00984     "C7EAA1CCE4B5B3795A0CA50DFDD8FC97E02514EF8B1116F96A0BBC99C6C746E63B229E9DD8"
00985     "F12B8F523E2334E63B497FD9E4559071F9FC5DAA1EDD430184902FDCCE250DC635B27CA59D"
00986     "6C25E97998670D26B8B5C",  // PRN 02
00987     "D201BEF00E5E7722A2C298AC291E48BAF9A23010418483F6E934B44AA77B2AC6A63E5431C4"
00988     "C511E6BC05CA757EB3B55EB419BC1CF0B9F36748B966BF628F6A93A25B5B82585919F076CB"
00989     "6A8CE44BDD5F45BFD61887A1D0C166E2E7E7072FCCF9153685F284864B172CD4412256CF3B"
00990     "3235935485D2D7AEA104F325EEEC7690E9128FA66FAE53F6F909391A1A214F59E8622D110A"
00991     "1B3B8F0F9353360F0D0089F004D3138EBE2242A2F1C3BDE35E11E0927835F64654C9C33EEE"
```

```
00992     "D8EEEB95BB41E2984506F055240EF1E4BC37AA0995725826922943 8F1DC2AE9B8D69EDF81D"
00993     "363476077884F8459E0530B154E180D5649F5F0976BB57024630B3E79F75F8D771C6A228A1"
00994     "F54243ECAD3F6DEA7FF0E8B2C00C51A7C2C57C4E126D1DE562F78729BD48BCA7BEDB42D926"
00995     "655FEB9C9ED27494F116A401EEF84AC1CD5FE0A6307E31CE8A5645BE4976CD69E892E7AF47"
00996     "F879FB74A1C9059EC7DF6583147ECE387C98F1D113056DA88FFA69823F1EC0BA85C15D1926"
00997     "5AE3B4E5FCC492D46FE95A41363EDF1FCE1B5587DCD85A5C38534FC7F6E8FC3792B07F18F5"
00998     "492A1EFE800E75ADB9FB2D95B7A0725A5DDA376D87195148958B09A8D68CD70C6A3DB61600"
00999     "01E9B905D9AE3F36702C31C599141271BCDCADB774161D89CDF69D357BFA138C0EF8A1DED3"
01000     "47E94964E515AF4110FEE5FB856348981DECA930BDFE097048C1675EF70CD2AE95FFF80869"
01001     "465EE4368F659E7D866CFD48939E10AD23BA969BDCB06B61BFB01089C961D69656348ED561"
01002     "1001726AA8F60222A676C45E64985C6D7550AF9A42242E84223EA30E40833C750D6615D200"
01003     "BE90DB979AB964E87C28D4E20B1BD77B483CCFA8667EB25B7689F7F0D6FD360612BD9883C4"
01004     "B124F1614B0524EE8F4AC",  // PRN 03
01005     "D4FD152565117A08F0E6EFB5F7F7887F38907B4D5AE8B678EFB899CEB429AD7DF00FB70384"
01006     "0BBEE662284FBA7C3F422D20BCF8699386DE3D1FCE841FF98E6D96729C5CAD69AF22A3D053"
01007     "64603D1599FF1FDC5541DB486E9B8616E4F0A6151F9DEB45CE1385822DE89422FE2E0E6045"
01008     "BD974469502FB22895A03162AAC82A59E8193D21CD4A56EF18593B61E26B4FA19DF9659237"
01009     "84C11B28D58FA424A286E74E0CE74BC24B1DBB1494C5C0D0689EF695272FBCF1357E01A147"
01010     "518042384253 90E2313CF280D93CBFEEEA9C9E4430F5201FDD172C46AA4F92F4E7CCFFD884"
01011     "B662D579130B5D292D4289CFA6647BFEF52850E3FBB2DCA3EC501B57012A1B1099C25BD1D7"
01012     "51428D52EABF52E41E4BB2788AF7F588EBA91093C89B72F766709351F9BCED9A0A5C00A82C"
01013     "D5B8395A847828682D2AE5A99F5D4199C0DED495452BBBC53D50CEFCCB92F294A1848C1625"
01014     "7A754DAC456F238E364B28A4C9C175FFE091E9FA4A0A48E7B6D82A4D1337A659F9AC752EA7"
01015     "73D09D45332459B3FC322BF89E29E6D8E8ECF2F7A02671FBB69A747163799E475A048EA943"
01016     "59C2646B7A7FF8462BD296A4A534749977A73EABD2D9B92C19D085554E4E91875989943E7B"
01017     "650BD52F279DA6F16B7197FF9F9E973075E456D7BCF81C847C89A60C9F772236E0411FFC62"
01018     "160D283645CE3DD96D143B99F20F6F7BF5331D7167E5429E3B4DB62ECC7C301E6DA239DD40"
01019     "0899B838A1BA42641FE3B5EAECB09C1D9190241C072FB1F7C6594CD8DA18053C8BF8179A62"
01020     "1CF335477CE97C05A6A9F7D63D9D4D651B23318285B3B7341207686AB801BD89DDAED3BC07"
01021     "B0F942C2AE2970136F0A73BC69455B4D312015141D20ADA39FA0382C6783181DB4380584FB"
01022     "AB7C4F1ADC15E261A07C4",  // PRN 04
01023     "97010A672C963DBD3D3ACAC38BA53CB739375BF1E093B2E64296D0D07476A0105C9DF0204E"
01024     "A54D9143B17756150A377A4E1F4F2A741B4FAFDB369B44C37F917F94D19EDF8EC3E00B7BEE"
01025     "325175687511111AE7F435057D4B7EE6F3B4512F99938F6F13984BC23A086DD96BB4076C815"
01026     "AC407D693E6FB1685C6D4B7611F3010CDEF1B2CFEF6FA7744437B8D199545720B19CF7BAD6"
01027     "A7B12CDFF2CC45811CE4C7A35CBB5EFCD95C2CD0BF16CB69C95A46B4541E0F7FB4645075FE"
01028     "4BC1E64865954BFB96187C265196CAD43EC41035774460FAA570186683A33E019719B14329"
01029     "128C4463FD32E5FAD23CB21D6224ED9634CEB7B0A36F359D3023D1142C10BBD85885329618"
01030     "E92BDCB786799423F38F7C871C1FB356C5E4008F9535D9C7334DF2D3CC4BCA196451BC47EC"
01031     "7B6C11283156BE86E53CBC4DA262AC1D19EB20350C5541F9F76AFF6A8F8055FACB6E1DD43F"
01032     "53AD0895689AD69D2BF12D9A1D1FFB14C7AFFCB0B9F0529122F1D1EC0BBFC582773EB58854"
01033     "7D68D4E494CC330065CEC477869F15985F8D60D4A5E8BE45F2D156C6353347270EF50EBD24"
01034     "48D2156250B6A489A8BED543561F099ADD1A205C1F32725DD2A6759C9CBA0215B06F93FF42"
01035     "3F7C10FD684E7310BAA9BE72BFDE2C2AD9210257E1C16DF9B38D94084F31C8E3D2627C1506"
01036     "AC041DEA65ED6C4E04CD4533DE37E65577F54E517781669A637E6C4DC98E73EA90D70A6D78"
01037     "EE8807E629AB100B68F2EC8282F232E8C8905D82FCC6E592C91A3E105C3EAFA38EB73E02B3"
01038     "7ACD8F6447CB22AC99987A8EA7CDFFB0EDA3B96C68DF4A4D3E18391C84C850161F5058802F"
01039     "2D32821AB7FC6B31C1013E4D1EFEB41234BDA4C84107B3FB5E7CA8D01816B8DE61C43CC1B0"
01040     "A9EA415DB41D4820D6890",  // PRN 05
01041     "9CDFD69656832C506F718657BA4A8EF7F0AEAEE78FBB829A1EF3295DC643FABCAB21344DB1"
01042     "DD8E1DF73CCAAF198619192AE27B42E470F1C7CF2907D1C8F6F0650F52BC4C0382530D1547"
01043     "52BEE42D38A03D1CFAB68AE474C22C7A4F4E06F2A3FDAAE81758D66741FFF25B6AFD529CF3"
01044     "C3F7EF339EFFCD3A3499DF2C85BA742FA348D449C167F7BF00688B11C453EE783623CC8476"
01045     "2AC049D9929D470037709C71BD6FE5019C9432C8A7CCB29904C1846586D7639B09DEC1F194"
01046     "9AE5C6D14CE92118750B0FDDCDCE61DBA6B1EBD92CA5B8DD456EB39936FFE010B741A22DFF"
01047     "11DC489FB7EA415FB265BD2A7361E7A428E47DE546103CC8E4E7F83A6936BCB1F9074E692D"
01048     "4C2E1BCE4C6065D407764469A5B8BFDFDC43EF9CB843952B5141208147C47ECFD4B5CC6953"
01049     "E77E0B3498A9E4930B9679AFA3227529BE29A8794AC7A23F6E8D5391B81FCB07F3E3082DAF"
01050     "B331B5A9A6C5C203BF1402458834D4B12F69F0A3EA8EEA9357A4D2A7FF90BD7ECEB441BAB3"
01051     "F5F6F150B542EC9DC17028E5C13BCEBAFCB5D2089D8F207CE493F65917022BF08E1D55A299"
01052     "F420D651610A0A7690A2511BC48ADA6F68077AB928A3017A01FDE99DE9ED02BDD06EB9C060"
01053     "02580554C378D701C68FC56F1C47362CF5113D4D728CE9B9E4C77B086258862CE67992B34B"
01054     "8810A4766B4F26BE34CE35A356E81A828735022E70F3ACE90CD175AEAE9479BCA317B8B629"
01055     "DDB6C7D29A13A0155E57332460466C6CEA034F79EA1A8345850CDEB29AA02701A5FA6ED331"
01056     "EB96A9DCFE3DABD9009A5DF5D4ADC8535675B073060FA15A424EE7375D039C923C9448DAEB"
01057     "AB6A881DEA6567FEDB235ECFA49406853F61F41D1906CFF5C343952F8E9141245862609079"
01058     "14E4C5E4225026034E030",  // PRN 06
01059     "EE21FF4082E6C3345ACE456EC43F40B2A12AE15C5BE1BE2807FB93BF95FC0D5462806256A0"
01060     "3C1ECCDEFB0686372D1E8CF83FD93BC5A4ABD65171A643C1E2EB2341BDBF5F1920659C23AE"
01061     "E9641712E4BD7AFDAD4E3E95CF019CE0F43E87B4838DE690D51B9A1CBEE45C8B8BCD5757EB"
01062     "B51A11A13652A4164269150A901F48232753F9C633D459ABA310880BC30008F5B8FEEC0924"
01063     "BD2677EA5C3073959497B2F9FA34F9A93D2EF528DA3B43E96DF37CE5C0D11C207699246CDC"
01064     "6D0F18DE1C643FD4FC60D80C6E7415920A75E5358B64B7D61327C6BB909187888 76CED884C"
01065     "8706EE4D7DE11E582DAA279C29DC0E828A2ECE0FBD336B63A4DBC07F730500F6D061D76DE3"
01066     "4796347839A13A286635F44A05C340B3B20D3FBBC48EE3B2E156BED9F93E0AD92C24205A72"
01067     "E16FB425AE3D8508D4D8EAD20080CB028B4000FCA39FCC13B0630BE3A4E2B2AFD3622CA1D9"
01068     "8D85BC88EF298B4A5DEAE3992D0BB1630A10D77165E0FDD2E757E7573DE3A17F3AF85C7B45"
01069     "CADCFF9A6D1909737934294485D54A62E8C681E64093562EF0CECA523784E1A46345EDF26C"
01070     "F69C895F5E5D3A1195F00D1DEDE55105666BA9819B7A20559CF624D5A0B167A8D63AB21A63"
01071     "30F89266A43DC90F712719302BD5E32AEDC6F0A206948B9E37E4B2ACA3542EE36ABD483B01"
01072     "00921531F2F9A6CD3135B250187C49628D52FE26F13F2A1C4E5ACC734C097D8692DF4F8914"
01073     "4C32E767AA5BBECFA3CD0E49A24F0470E482AEBAAAF5FB33B77B52C1579B07A9D22952989A"
01074     "B1CF9167CE3552F6506C657E93FBA9A9E9C30F9AB133A01C8CB57702B8DA67D64D6BE89ECB"
01075     "7D75C431AFE28C02FF57FAC04962C412718CF9B8740F42BBF580A7DCF37E72CF43C3958D5F"
01076     "F7756EE1930A9EFC7A91C",  // PRN 07
01077     "D57C59619A6F398CC2A89D6D54F9E1F846BA7A7DECEA83A16101DB4BB31DBAB22CA53DFBBD"
01078     "32AAD3FCCC964A4C441B6DCCE145BCB7A59DE9AA123E8D590AF9999C7414333ADC7BF0C2CC"
```

```
01079     "3E3461232AA7C4633B626BDB6EE99EADCCBAF3D2F7D51963A604C0E234B5B1F19F56301C24"
01080     "65974324FE52F22355B7E8229DEC199012AAEBDCDE7DA613C04C3D978D615C07214E2051A3"
01081     "D8BE099AEA258AA7765FBD3E9BAB6F572B81F0A6FAE0C2BAF0672052ED8B23AE3EC6946899"
01082     "68E3E6F1055BD70BB78E82503A3CE3DB48EA5E47D229E976B0C48CA00C771E377F02CC0834"
01083     "2EEFAB06F7F723B910B14C6793925C11938756A787AB0A05E123CEDDDA545D70FAB1E78CB4"
01084     "053AE3ECD76B2021F5F9105DB0B3CCFDF0A9C2F4E5486749126BD7C4DE65015F84702AF1D8"
01085     "A4213E5C7D487379C54FE1E1224C2D576217ACD1490C3E0FC84B24A2008C55F335137195A9"
01086     "AF9CBA46F695EE5F97C7C3B2165B561920C1C488D9B165D1FB726B63758B61B8E93F3E5BE0"
01087     "97110612DC469FEAFE150E67F594F2B8D5BC1EB4AA67F733FE309D96A14C0716454DE41D2B"
01088     "D8668EA6EC1084A3C2A0C761D78DFFBE70884D7D24FE54AE8DAA71C857350395CF11AC3FA4"
01089     "BB6AB84F58F9856D0802633DFE2C67840AB432E00F42007EA46AFC0CB22AF6C97241F8B7D0"
01090     "5081AA67CA16B1217D467CC0076C8EFE15B9EAC83E08D65E13A6686E8F72988343411432DE"
01091     "B668A82FDC4B73E2C1CC9086CE058EBEE5726C956B927D45FB646CE54ACAF821E993BD3313"
01092     "2FC549C620346B234AD0D86422BC0D833F8C866279F12EE7DFD6C6F7E2B088EC6711C2CEF6"
01093     "EA64FD758345040D55ECB443BACBFD8AF4F23A1703A8B1B633DC5176602D1A842574AD30A6"
01094     "4403C37C03BDD026C6354",  // PRN 08
01095     "829CC3C571A2BB19164855795A60833734868AFF6C47F933119B4192048D740AF94292F56B"
01096     "22B263CAC2047455A46790417D2F8170D05C57C62CB0B06209B6A12405F7FE9146EC142A4A"
01097     "B8F7C73768A4E98469B897FA1BADFE07763EB95675EA202BE998BF04B68BC6E3ABC749F60A"
01098     "A44DB45C609CEFAA011E3748FA1FEBD9A5894C2CD399C644D09ED9E25E8E574056EADC9670"
01099     "8887685ABAA4CE56B27D7C7CCEA8B28C4D8A9EF758A502747717C7E14B8E4DEE68E2B545F9"
01100     "BB5BB94E53AE779638EC429BE9A9396666BA6961711DF385FB4E1E9106BEE3F80AEF8F5392"
01101     "FB31E74E16168FCE9E1969724576B522A4C91620A44F0EFFBE0567E87C7B681791D4E51006"
01102     "8AB39D7F16D5E78E17EE00C14F722E4CD48AD67EE110401C70BC2F9F0F61DD1BC91EF6741B"
01103     "3A77745EDBD43BEC8036B6E4A2CFB84A5C191EC35B561044FDE81A7ECEFC4E646AEDEB9F10"
01104     "E5303E2A25AEDD8F10B938E8F580F766F73E2A53C04D7C82DAD64BEAF4D8EE3A47943A6CE5"
01105     "A2A3185D5ABF2C4B829D02F86FD976895056869B28B36851337F36412950D8CD6F957EF06F"
01106     "7BCA0633EAA13ECC7E600E2A1B3C42067F241D79B07D045A3E38990F6B6E0A44F2C10834ED"
01107     "3A2DE1D172534C31E8ACC6F66D9545906AACD083D8EA0CC30F535883D6B110B1CFEDBED33F"
01108     "77D84F698446C92124ADF0B633B62E4BEA52085AF0AC40CF80803A0459E345E6BE44D925F7"
01109     "CFBF10D7EEC73816C8D87380D93FBE77C300B2368C2AC45BFE5004AC6ABB4E68377661112E"
01110     "F717EF4F34B6BCC382D0F08F75799B2451038491A339A0C5675DCDFA5044B89EB4A423F1DA"
01111     "084AFCFC32D9DFA6B7808C6A1D108A33B8FBAB59A758E4E6CAF334971413066908B0B1D56A"
01112     "373515E8F1EC066E8AB30",  // PRN 09
01113     "B2413CBEEE7A6CFB5400DF607C16E694DBFF1375B9AB755724EEB0759221F2CD261F15C2BD"
01114     "0D3C8802B5B035B4E9F7976E3A31286A9789549297DA272693746E2C3841413F328103F0A0"
01115     "ACF1AB94F0E35626F11B5D8934BE54DB9524F9B5F9BD4EBB46BC8350E78E4C05885F7DF3CE"
01116     "D09369D44D4025DA9A861274A10AA161D4A69701F03F1684716C6EA79FB19CAB0DA4368FCE"
01117     "3BD177D2B9EB4C9729F615AEFE232B7875F2D1125DB8838292CAD93F45303AE0C8BFEBD050"
01118     "F32A9F2BD11DAB69D84976441ACCB6311DF1AB09143926D138B336B839DDD4FA5FFA32F1EE"
01119     "65F6F7D2FFA363815C288FBF708F9D81BF4AEE2ADE4FFA5F2FA647EC8DF62D564C41A5EB04"
01120     "BFAB5803EFE44B041B7A2798AD9EF34204F573D855136442B5AB6A85A402399C1E4EBC0D11"
01121     "1C9C3A374FEAE92B6ECD22BF21A47F7096C48846FE685236171B4B19C80778F4236B68078C"
01122     "7CC3CDFEF7CBA3A7681593F0FADA37D709598AB0A6DEE2DFB0ED8894D2457CC268D1C4D78E"
01123     "7FA50B99B19AD4FF7B97916A87B39C2694C24C360EEA1BB0920C2DE28AF4D76816E63690E3"
01124     "8BE3E85BC6047F8AC762D85C0D641F15497E4615B295E314971EC646136D0418B42D0F326F"
01125     "B798407077627FAD436271917398FBE18FA095BFC7521E361683E464D2EABB64D5C3BA927B"
01126     "9A10143F16C528C91E50C625807717648B7396EC6BBD820FAC6C3C30A517CA3A730A4C7C38"
01127     "E9280A27C6496723DAC59D8F02D0519C306069FF2F1F6871892613C11A2C7001EB913341B6"
01128     "F0A9B2049CF8F156B695C10ABBCBC495277755FC3E824AD7B947F00BECC85AE53C5C51CA50"
01129     "7046B78E674D7CBB206C09183EF03DA63D911484B790A5128724557EF769B76167C8126D43"
01130     "32F1C42EC16457EDD1108",  // PRN 10
01131     "F4C4B94772CE84695325CB3AEEF60BD564147ADEE80541C6FE5F5653DFF51AA05764D6FAC7"
01132     "B59B0F365EFAFA4CFA15EAE9AC57D932F9B73EB90328F617C9588A8CFBC92805A025EF87E9"
01133     "2CB06A1DDAFD24097395C4D45189D229B793A241C80E4124F9890E018647BEEFA9534B5118"
01134     "DBEAF65432D66D2A0A9755F5EEDE57D79930B5112F4823632F375A9B085AFE938D862139EE"
01135     "3034DF716AAB84D9A233F82561426D33BF33D70C3181477B24BA1C3E70622817414C58BD09"
01136     "C0CFEE8244677F2B3F5F12BDD5CA3300CFCF23F854FE08BA140C22E17200D17C4C368C8CCD"
01137     "A00B14B289F91F75F7D44D3F556C26E8C5476340DF71B01A993038DB68AD589ABA593CEF78"
01138     "852B567AE047B1776386200978E45FB6E7380ACCE6575A12EDC02A6F4070B570B40CD05E3A"
01139     "16949A681771A349E66B6C8BAE36C24C4203662E1C1C0C4744800D4E0A4356CA79DD087839"
01140     "D35B3B1684B1FC6748526854CAD7EEE517D54B4D682C6951C74ACB98E8E63ACB1567585EDE"
01141     "5D056A1C375BC82BA4D9E3D1773326A54617B0790E1C687C15BFCCE14C8E7A84670C01FE0D"
01142     "7CC7DFEA604A644EC552D72AF3BB3FD3F4CFD4A94235E132165D14B1F412331FF47D6F7AE5"
01143     "B2A429237B0D9C2085EFC7C0EBE3B799D78621D8DA4CC7A67966DCEDD37AD84F9E44A775A1"
01144     "F1CF610077CA993E13117A2B0868A677E4E677F548658D9AA5CFE91221CCC623B08F65F0C7"
01145     "301A0877E97DC3A6928D8B8E69E702703ED7A760DD68AE184BAF4E9F50AFC38138E52FBE5B"
01146     "57BFDB56230C4F4D14FDF6A95353E26A1366FFA0B41AA3703770D46D73B3EB27603C35E36C"
01147     "008497C3CB01D8398E5823370D18F40B3BAC32ADED06819135A8BD06755963BC7793E6E0AE"
01148     "0AF9B60FB948E5103302C",  // PRN 11
01149     "D174C617556835B19D5960E23554CC72DB74D0EA1FAFDF8C0246CFAE6A8FFEE862CB9A5281"
01150     "EB8F140068D986D133AB150D40326E2FA87BB3450DAE0804A4C62C9492F5F96553D9D43A1F"
01151     "AE170114E9C445302E9D21CFB0C19A9EA0853A79C49379B40265A44E0817212CF5249016A2"
01152     "7C4C2CCE7EBA1CA9FD688F46C7529ED9706420A3ECCD1B5057606BEB0470A3C763786BD777"
01153     "1CE7B9FDB5C87C4BA27A3A7E165036EFB7F0695CE4365591206E27EE874282043D513A920B"
01154     "9438A1F05DC14DF93C5D49D07CD44873807CC57D75382A2C5E6BE193743B317D5B491B9984"
01155     "EC215EAE7B9426FB4F64D1C6B880635EB5067479C9AA351E8D6CBC98A65A12238EE347173B"
01156     "FCC47E6D8ED753F4E6942DCF60A0A4C15C21DA17A13FF00A50E175F7935EB113A088E4EF86"
01157     "E87FA710DA93B99E99ACFD01FEFB68F4154AF85CDF7C66D20355F9ECD51969CFD33ABCE58C"
01158     "1B1807E04212C4D0209ED76AEF40116F861BB558585E144C9351451CAE605D0B765C87B5AD"
01159     "51A7A430A0B8D5D05E58976137D5102CF771544AF66201B253D4665F1FF69E775A18D4AB7F"
01160     "24A000A3E2375694B03CF933FECAF701949BAB66B51606569ECC2EC4F24D2720E5E1DEB68C"
01161     "37594A1256EE376EACE92A5B728B9A0D1EF50530FDE76D94DCF0F83D393E3318FEFE11E035"
01162     "23BBBFDFCF4560E43B583139DD17269E87EAF57ACA8E202C51FD39DA50E6095DC0F73332CE"
01163     "D270DCC09F1A6D1613FCEBA8E94AEDA15C36C1C4EE6473C6688FFCDAB46F1E2657F3108452"
01164     "88213398EEF25C72E6FB17E9A375471B96D5137E1246D035B6681371BE08FAACED63444E9E"
01165     "36BCFF48A69DA97FCAD19F4A575C8C8FC7FC4ACCA11DF09479BB8488CC91B9E1B0C1D458A6"
```

```
01166     "B30C8369823703C20E29C",  // PRN 12
01167     "C5DF1B2B634C4D2A78DF4B5B3E241E680D6C23AC76D6628F4F54640BF4A02635EA662C21CD"
01168     "A94970DBBCEC6FECB55991D712F0CE85F1A51211302CC584E53528B385CF87244ABCDF469B"
01169     "5904163D13053885C3FA5362CEC766EF045E712039DCEA042C05D4DE5F40D96F3A5692EA71"
01170     "0E4BBF55E343F0F91DAA8B57089637F5660E0A7CF3A972FA5EC1F3CB709DC466C9C6947F62"
01171     "5B176F7001C1F8023813593C71E8525538BE92985638E24F65A76CBA45EA7A4A91A649D012"
01172     "5BA25F1CD3298FFD529720206788AFEA0069BA99F2C99E5936870528E6050B3AA4EC63D152"
01173     "406EF5FA6B003E87FB513FBE49046ACFBEE51B976D19E4155979FC8444A215799774A2EA25"
01174     "4931F21B2BA11226C589029898B057EE6AB6FE46D835A70382581AD5CBF10BE6BFFFD3E742"
01175     "2D92688D6B21A1A4F0E6EA33B75DC807ED6A81A1C680A0D404B336CDA9C25842B529D52CD7"
01176     "C47199CCE9025A3376E7A862920435CA63A9A15F2205BA89F1058E11967546585FFFB6D447"
01177     "E608DDA0714AB130066C6802AA85749F8F04D56266DFB1CA1C14861E789F1CD517C849F6FB"
01178     "ABCC77775501C4E0E90EE3BF6DA51F6B601040CE72D9F5C5777B2A98F361E587373E3E6512"
01179     "D754084CFE93AC63350FBE72C7C0BDE2C6F172F9D37035F7547830362461FAA94EE864FF63"
01180     "AF608670F75D835F34713D23724C538514B5CBEE32F5FA38B5ECDBDC38AA3D76F98E8B198B"
01181     "FE357FBAA9AA4F4EE2B2A6A6359B433C561FA6085F31C68A597FF3F0264ECFC0F60DA72761"
01182     "3EC063B7C3CA6E6B44DE29BF78101BCAB6678E1E3EC1F1644768EE11BAA498055AFE6506F3"
01183     "71328375D7BC6499E680402E254FBF35F13DA867316A310A67A99CE1555BE880EDFB23F611"
01184     "697268C2A7FE22AA8C484",  // PRN 13
01185     "99553082DACAED3A9D8C2BC62B100A611FB9290BC70B80CCCB3848DB085EB673D54A3DA91D"
01186     "7AFF3F32104D82C048265976A4D434643895328DEB977B0B443C5B707874FEEC0F4F2FCBC9"
01187     "D11B1533D69F76BB0E0385B2D2C63B6353A7A0C3AFDCF53FE9E114376740BBADC000EE5D91"
01188     "AA0374537D89C52EC1509AE47872F21288FA9730A287C6DBECEDC910C04488789093830392"
01189     "E19BD364B128FFBD247246E12325C1205429E6742D0BBD7215CE420145F72CB29BD25A906F"
01190     "5C798FD4A575E775CB7D3404B7C210B0D52E035841031AA8DC341D8F51CF76B5A0BAB7A2AD"
01191     "485B2A35E024E257E438E7D9C778D3D6C4776ADE71500581D748F8678682A06A92AF0BF58E"
01192     "DB721827B4E53CDEFCA2CC17D891C5CF70423A2351D30D68249275A12704BE1198569C06A9"
01193     "A80942525F559AA1DA963F488EF43BC365F9851E4BBCAB743BDAD827B7D0651C832B1E46CB"
01194     "6A3F86E2776C28B91384519BDE28B718BD4C573214D1A5C612AF7766CF76C62B6614F31F86"
01195     "843019A87EBA5F1DFB1E9F75EBD93C70211C627030BE342C81D872AA48B819F722D401413F"
01196     "AD3249D21409E6C75CE3675AF1A58AFD6B6AE89E187E7F996AB04C8806FBD228A1BFAF3C96"
01197     "8F25DC38F5D5A6BE70E964903CA9E26490493B2CDBCFA21EBC06EB4D091F68882D5921CF46"
01198     "2B633DED2B22F6C0B5BA32BFDE473AAED4438EAB2702A96D9521F2E0F77E6A0572245FA7C1"
01199     "CE8D208C80631065B742CB54A960CF26FBC39BE71A3DD8F6917B5E42FE004B240DFF1FBD3B"
01200     "EFDE62D365CD90DE8C8E6EA10C0BB41B2F0E5E727298A51E6E17D7C1DB21463D7A7EE7D31F"
01201     "6D6FACF6EF6BE5DE2923196102A1E333E66650AC7560BA7BF414FE8DBB6BCBB7D9984FB2BB"
01202     "C67CFE3E429FB0D89B118",  // PRN 14
01203     "CB1C401CAF1A29C20B51EABD89D7D33CC1A87FBDC6585DBE08DB43E71263AB39FABDD384A4"
01204     "A3FC2123F946DD41DC71A001DA565B7461298E3F7563D8ECE03A88E32E0B160F7FEB1BE347"
01205     "8EC8A5678EDD796C4189653F60B54E1C6883E8C8AA39E25339C7ABBEF64F44F0A101F1C0B2"
01206     "B2C9C9D8AAA52D6138A97474C6846B748A00C79F0E6B3199606C96F27EF514D0DF0A7E19FA"
01207     "B0FB6F2BC285E29BACD339DBC0DB249D7793583CCD255438D6D6AB60F4391E5B00432DA093"
01208     "7DA4E0C5CB78F2B3260C9E1FD4EDFBCE8B58AD830317492CD55482600E960AB8A817E273D1"
01209     "40B98AD2A65D2FA49CEA1999ECCED6C959D9F1E2AF88AA2DDD5CBF23F24E97EDEF4848AE69"
01210     "C8776EA1D26005D19A8D5775DEFAD8D05D7264290EE8801DA11CD339BDD19226920DFFFFDB"
01211     "CC9AEA51DD307C46727239EA1A75308DB6BD265C3385F86F8A508BFFF721EC83EE506410DF"
01212     "67A455C394F53CD878E2910DE68FDCCFDDFDD1BACA4C9B0F88F0B48069053449DBA65C88D1"
01213     "9F8933071BB92BE7A8C90B5C3A273543B037D7376B2E62DFA057255696694E3BC4841D7F60"
01214     "4D1AD887977EFD806C352A8521C87A170E8B6A34B51B1BD7A8D100AF8B28B0688F236F3E46"
01215     "CC938DF606F971B0220D1B2DC5732FD84B4653917DFBF880EB6FC4CB5DDD96783513EBF955"
01216     "D6532097DCE92A6B1EF5FFC14122C3487B250D4760160DDA65BA139AA6C6B4F153A563215C"
01217     "075A6FF3F69F91D20B19859810D090B80485F8A6AA7AEEA92A2821D28D30642E9AA09C7797"
01218     "4EDED3C5ADE25381C17ECF1B083E282FEF30C861506B7570A4898A1BEE9FADB6BD8E5A1D76"
01219     "4C240065B1E20E12C880A28F40C70360228F609F23752BEE6FA3131568A48EBE3D3A233D49"
01220     "E038FF9AB4D38F5F1C844",  // PRN 15
01221     "B84F109CD8389BCECD75039A199EC00D2EEE6B25AC377D745F53A5D1ED39FB1FCB4F6F4AFD"
01222     "777C4A25796488CF472B401D92CC72EA350ABD40F97299A4FF136DE0279E565B4A12EEDA83"
01223     "7FF7B19CA1E93F5A99165882444D13C93CA89E6755379E20F625C6100E4F49EE42B3C0CB42"
01224     "DFF0DD0D48B1A493BBB70E63212CCFAA9AFE196E2439B464D8CBA5C2369DA2EF5CC516828A"
01225     "BA5D913D8073C642A30766A58EA9D2EAE0FDBDB7F53EB787BCDD5A2D39A0CEBE2CE4821378"
01226     "4BB8AFA96649F0464449FEF244134754AF150891B2FA6A91F46B6AA89CCA338BF0A24C7D32"
01227     "C9E65249501FF157C3882B6901A92BF0F8404083D4E303BA691B8C459A20C28C20E3EF52E9"
01228     "B723E3E66C83C517B3546568C18348B0F6A6280FF96BBE767D26E885122B1969017FFDCCDC"
01229     "CBF9FEAB42704356A90D45E51F4B831BDBF9876D3DCD4E9ED85518DFE2770ABEE905255839"
01230     "9FC54CFA8F681468D7E6BC6C86B041A0A076D99C8EE9419E2FE6441B260D4D1ED17283B12B"
01231     "62A383876A4661713552D56FD894FC6E7DC47BF5186784DF1F94C777B5172767EE0BE98178"
01232     "334F3CBF35BBF356095BB1E0362D5CB4E98A90DBB9388C5799923F13BAB7463F8056E2721B"
01233     "D1FFF68C157383D2BFC00DA77BCDB0B551BAD8A7F775DA4318A27D64BD8B83F425D24187DD"
01234     "01751069975F55DBC63829CD1E735CEE21FB56F5017C305C359339C487B0FB7E58EF0824F68"
01235     "5AB6690089EBF304927A4A18601DD705C44FA55A42BC89E12DED3EA505067AC0F8ACB14BA9"
01236     "607890915B4837B1F9FB41C14F2016E6049DE6F6245ECED1A3578689CF8EF22096042B489D"
01237     "7C5893C65C0D2AC101FA9057519B72562E724C2A3A30389E743D3F45272064B06C13D8B131"
01238     "6A5A660326F34156B8640",  // PRN 16
01239     "B42ADA523F329F9CCA7199214DF87AC1C50118F7E238FBE3CE9CE8095F79E78481602141FB"
01240     "C3B87CB1F7C68DE22D74F44F4FB30B2EE48F218117512185B3F1F4AD2EAFAC164E64B5DB1D"
01241     "D1D0425A6331638413ABAA7940CCA0E6C0C509B0379CC8A36DDE4AE0D9A66A2475C4E9013E"
01242     "62DDA0A923FD6427F119E9F82E3E75C580865D23F04D99D2BBBA095898BE9D923E68E80C44"
01243     "AA0BDF2539B2F55ADD4B7C9279501AB2AF650DB22494441C5C5CFF830E0AEC172971C5DF39"
01244     "20F49B20A73E4CB5427217DBF6165A09A038F22D958015586E4AF9A9ED33BFF2F9C72064BA"
01245     "8F9AFC92F4DB4480348C578D6D66127C3FA9C1BEBC210129F18997467E69F3FF46757B0820"
01246     "D29F714D5E58E67D0349F1E8936EBA52A9B1617AEA064B1CDACA91276C51858FB8789E9E33"
01247     "F8587A732B20A5FDA984A6BDB86EF7A8E26D56380555E8CDFEFC5264962B4E31DA645EDB62"
01248     "DED96EED3E495A723DE2F80DDEE78A8A751DC24CF05BDAB15A0E4FD50B311E45EACDA6D1DD"
01249     "4A2FE6367AFC07BB54D5C040A16EE0B892CAC0CB791EB9CF7CEAD2483B9EDC0E91E2F99573"
01250     "0318B226D663A699BCBD9AEACE57B842D4890D9D681429A8D382A21617242E3871B39D1568"
01251     "1BA8838BAFF11974DA1ACAACEA54549800B6CD1784BFD9445439D7F1A832D4278688E77DD7"
01252     "E3E2A51DA7BEE25BCC3DCDABE90A3E06D6EC0D57F0EBE3C98E751567FF5E1E0E9C3D74DA40"
```

```
01253     "E49855A0AA4D35D63F5BB1901924F887C563951D144B75B776E623EC065B77781AB0225875"
01254     "1A06624C5C803C6BE60EF6F60D24103AA100CFBE7124CCFAC338421D614AD80BAE9CB2F80E"
01255     "110E4049085B681DCA3E44C8561E548916F0BE26B967E127F4CE9BCC23BB437BC85FD060BD"
01256     "AACB1E8F2EE47AD3AFF48",  // PRN 17
01257     "CA255F7D7F404B16F1FC7911712420551AB5B3F9F79BDB69C726E5BE2D99D6BCEDF3BF5319"
01258     "D2692BA47D5D043020EA4D53D9B0094D4B9B75EBF1534C19399B7F4B61482B63D8CE0BD3EB"
01259     "523917820D7655808F39D1D33853CCD1DA81E21757969D497A8047A2D7B1E90B969E13CBCF"
01260     "FFA528619B83698A1E9A61801E430C25B18D3972E103647872C252EF4595DAA9F9D2DF35B2"
01261     "491E3F8CDCE6E2EA7D0610D09E4D5474CBB17C12F0BFC424424B448CF30A14C80886E97971"
01262     "61C716B7AF3F72B02CCB7E90C46EE4BEE1F85B2E9C950B969DAB140F3728882D5A162932F9"
01263     "D41ECFDBA16F652AFFD0CF6CF6891A718E9FC2FE0B5AFE92106985B42CF866EDFFF26A1512"
01264     "52BF5CE381DDF779FFFC75E7813898497DBDDA3555D16809086A0505F831ED47932120AFDA"
01265     "68D423FF4F62CDDF66FB84BCFF787658C103379DA2F2DBF2868D8C436E3D555D148B6C51BD"
01266     "4E361E8FEC932CE4C4D204FF8F10F936D1C780D7603C1824C222323E5D32ADACBE5510D6E2"
01267     "AED520CD11560AEB9367D22D9F29506D407078B9D8AF8BEF3087DA339C9BE1412F3C16B131"
01268     "81BDD8397D6C84791E9C3CE73AD5A2279515558B7FF69A0CEAD79D3890DE3F4D89ADEE6A64"
01269     "6E01D0B5E45924E08F706642C6EE18A3DEB0044325BEC020B12EBA41EA540976962ACA2ED6"
01270     "28E917E7F18596BE3CCC457DCA41A35A0499423C643BC3C95AB34C4AA2BF65E74079DEA268"
01271     "1169DC3DE6DED27E7596C26ED0614EB77ABF2219CD294C9C4E80E09914038381D2F8B6D472"
01272     "B770A0FC860A0CFA2B27663A41E6354C45AF8DC90E5D12F28AD91A3200A626CFD837C563FB"
01273     "FDC4C7B0A8319E8C4A4975CE54DF744412BDA3B1D54C44E59D24CF83EBBC9B6990A4B23A62"
01274     "729500250184E63E1E084",  // PRN 18
01275     "CC8AE040A281117C472F7C87E108B575563E252A5C96064F9242F3D7F2544FE06DBAA94F68"
01276     "47BFB89B3B6AA1C57F5A4E59FEDA18E365A24F4D1E3A8776768742BCA81BF7FA596A7BE09C"
01277     "B190CF9FC5D937FBA0CBB6721B3A8B6860AC3EB8B39A19365DEA72D22DB146563106544DD6"
01278     "E4EB47C7DC740F2AB1C5E44660D41ABB467237FDAA712BB7EBDEF277161178483A3E1D73C6"
01279     "8E42113070E92F36C82BE02048B61735CA16AC302A0821BEB0EB4A4C84E425C61A54955674"
01280     "4942CB0F6055ECCD3D4D56B1610BA7C6621C75AC449AC02DE04A2770E792B65D004AA9CD8D"
01281     "FA96F12C5BDA8782FB08EC0E3EA957CFBFFF321D7C57F8DD78E4D561279AC33E522E33F517"
01282     "FEB5BB78ECC3BDC325E8C5FD75BB4A7362D9D995AD4B86D3DE969D06A04FD9C99F3DC4DA88"
01283     "BA4F99D1C37C2E96A302CADC30A7DE63FC1F75853D2E8B15E2396CD6617EA77129D4D63EA1"
01284     "D103DB0604B0153F234B48EC4F74E312D1582CD88CF6CCB958730106667A490DA0530BB66D"
01285     "909FD1DFC725D104ECD8FE1E8C3F0ECD9553E3553AF1597CC690BFD9E83690EE7DEB4EDB60"
01286     "6961A1E53716075564251A689549AB1CF26C14375AFC82510F8E5EC467EFDA137D77A3640C"
01287     "87099EEE6052C89356C1A7F3AA2274F6A7E6485C5EF1E91EFF12202DF45C6F3CE0E7C0424C"
01288     "141FC946B451A472A9139AE0E9EA500FF86F3CBC9EB021706038E708A32CCB11412B421045"
01289     "737958216135D63B7364181F4CAD70B7C15D4D0DD8DB5BF2E20A5F103855A7AB306F26055C"
01290     "658308036A72637696987E5D84B8664CF552FE3CDFB86421015170CE7F802DA7CCF8921096"
01291     "536C3DA9237ECAE43A16A736DE01C595259AA18B858ED081183C726B63FC2D1372FEDAD7F4"
01292     "8C1950FE1CFC155E610E4",  // PRN 19
01293     "B7BEC67BE0FF68762DBA40B31D235BB9361A98F8D2D538FCEF76E5286EEB677CDAAFCDEA26"
01294     "8F0DB6849E140A4B0FECAFD4D02927EB095437C2C6C2B2C065FCCC14CB580A647CF53CDDF0"
01295     "9FA0DE8E5DB1CD030EBD0CEEBFA2FE7766490D0C61A690DF9AAE4961268E51B8E5E35F837E"
01296     "F080BB64B21DF3039B36A621432848408E3F7C066463352C21FFB88D06A2F87966FB35B65A"
01297     "8D91BF29E70B3FD78F67C0998BE8622ED3C48EE6112FE4D6D176B448D5D55C52B60A1FA613"
01298     "4CABA76E5FBC74CF4BCE85A1B6DE091899B7DDAC66B353DE01507A6B6F69E75A20071E94FD"
01299     "57F0C26D48EF70F23C735AD1C21CB122709E79BF875E150CD269459EB5B8D328EDF5C34261"
01300     "5965ADEF6DDED2AC583B1F7834207949EAAFED3E1C2CA2BE46F572DD6A344531B90634299A"
01301     "09929C602A3012ECB0738D36B9BFE3625DEF26E11F3BF179A4ADD324AF1EA860993D4DE881"
01302     "A4D32B209432DDD7D083AE4A37E3FE25FB7CD5D47430311D37B93D7F3191C91ADC0534E670"
01303     "F5871163C09ABDCD72C0B8BB8697C62EE9BCD3169066A97FBEDAE909327A35400FA12F8915"
01304     "E546C1830898EC814D5BCC459B24DF4A0C4E40AF4058241DB9897A3E62142E92A1FB1FBF40"
01305     "B22ACE4A1F14B745411443B5A092AEE0539374F6DDF2FFF84DE33D76295EF6E2E8DDE63943"
01306     "5821BCCFC21B39E034EB0152C2B5C5C3AF8F073A4A83556F6492B07C86534175BEDE22F414"
01307     "A4E77012A46A789822A096246AA08880E43839FF5D4274784FF3E6240261A201CFA69A4385"
01308     "944C4E513615BECB948703BA86E36570C45499043A457256B0973268BDF6747F584F768705"
01309     "6FD7C8060F2DA08440582B6325AA1DEC779381A157D267C163AEDC81194D415B2B041EB151"
01310     "D7EE95286D3F53FB86130",  // PRN 20
01311     "F4CF29F794CC9CB418F513F3E8D26E34073939CAC74CE0C82C99505DE0BEF392D3F95BF717"
01312     "6EADE5D31CFFD43977FB31FDC78884447A3B7344DA50CB56B5B47A0BA9791E27A45F41B474"
01313     "C99F801996FE467FF80CCA7AC3B1B0BE33265BF82C4E88D2F660BB92BD75E34786977CEF8E"
01314     "8DA9571FB56515ECC6D3240DB2A6FDFDACB2A273AD4BDFF88BE8509AFCEF967B9B57CFEB11"
01315     "5B06B0455A6AECC65C588964261104103EB7A78A5CE531AF579F8CA534898348042B14A238"
01316     "C6FDDAF2E5F16FF9B204525CAD0C3DDAA4B8A890D3BB97B963A0C9F8134DFA0A21B6195857"
01317     "663748BA48C2DBD2AD7E4DC0841E875B36892BF76144A5C242E969C19421C18D8EC219AF20"
01318     "6A1349AA822E49EADB1E97DCF852C3B3A99AE10A5F090B0976E9DAF4F72C1D43C72F39AD76"
01319     "715F80CB472B2A15B7FE555C74D6B808C1648D7377BE0E218A3DB00B405D6396088DD457F4"
01320     "72209E387BB9F8C2A941E0C1C9A28D8B3FA18AE509ABC9F9F2F727C5698EE520AFBD786BF6"
01321     "668A5A8AEA9F359E27AD48CEE9FAE878F2C91C43CBB927B761BBFB1CA183DA0A2987086140"
01322     "C8241F5D37A7B304B3677E5D7287CD7C8AA4DBB0DF85AB25EBA59191E5CD22B06E389B2000"
01323     "2B848BB6BC02CDF28EF6C985D048FE3BC88002692523800933858AD2801701C9D4436F35A1"
01324     "627277A6ABDD995E85A4620A940E3F9E56612BF25F3A0E17B5AF7491F475133667643A71F7"
01325     "2D75613C1546CA722209114545C8DBE2CC0A1A57098F66BA20B3E674521F417D0716EB4656"
01326     "1561E5C3CBFFCBC94A7DCBD15B140ACE1DF47FAE637930FB665520A20323DF7C5812B9308F"
01327     "9C3106CE1F84B814433A54F50243CAC9A3D24B29CBD981191285E4ED9E238F91B04600B79B"
01328     "2DF3B6AE4188F7E60302C",  // PRN 21
01329     "D84B4EA8E6A13C1596B2F0266D32A11425C0BAEDF57E1DD9C081F51A3F5043C49FB2D4DD58"
01330     "1C8F76DCE746C2D4264371D459C60A0129805C7011CC57B05ABBB743F2EC90F7278A101C44"
01331     "79D176EBA5D711E32189B11F6CC2419A2BF659FAD2CCD7D67B51566C623979A2EB8075F5E8"
01332     "EFC6E3A18CD9E705E73B948B1551AFC0AE902353A1510E7F3B0A4A446EE19F61DE93AAFC05"
01333     "53027BE4EBA7C1D9C76575F4BCE1B3C8620FB781CCE777E587E587F26CB62E386EDAE3B0AB"
01334     "6CAE5BC4F7149F183AB73AEEE44ADB412AF2849AF8C9D4AD6E9851D99FC206DBC2989FB554"
01335     "072810705B57D3C7529BB38D5006D24014234D285C44F8246262C4FE9AC4FCB3FAFEC40DC7"
01336     "51BCC9AC3CE82E3C3A69346D08B7AA3902317582E9626402AB299D9DA83D3D43EF13FDB6D5"
01337     "343C36F04B9250E98481FBDB766B81D0B669CBADD4CED277D69E15A6B1ADC24EFE4F47633F"
01338     "AC44F80D53B85BFFBA2301A647C753480AE5150ADF6854E79ED71CEA29A87D58E39B220AC5"
01339     "840F1BC0187635673980989FD08F3EED4538B6ECBBAD4768EF211C9F5176D013545F5013DC"
```

```
01340     "9CC786FCF32D4DAE3C0D299940A9A3109708277E7F0A93A796D3FD907811DB4E824605600B"
01341     "B21BB1A53A395F7732947BB0C68506E830819F84245A162151461DD9E2D6C37278413FDD70"
01342     "B0442179B9CF44A7C360AF5A905E6B911AC253B0A3F9CE36794CCFCD4A3306CB0AC11A8B0B"
01343     "E9320C1F5ABF1F8A9FA7221ED0FE3FC69CEBA0D93EBBD7B3BC663C3EDD30A0290F26A369E8"
01344     "ECD629EEAE51F93F43595B833294427FA0F5D14A4D8E7B116C6B66C0A76612B09A4868A828"
01345     "66E8FD4C760EEF4146DFB2A235DEA57B39877E84186A0EC98A095E977BD2D2C444BDDA8672"
01346     "08924B8EDC5D7D7A65A54",  // PRN 22
01347     "8642BB604191809623A899925E0E67D0521EFC720D4C80211E7B7BB4ACF2CE3644922F59A2"
01348     "6CB2584CB354BFC2AB83790C172FC1F2E8A01470DF0E0542422EBE3512D1A6526A439514AD"
01349     "7B865168795C99D75D535DFBCE2CB4E83AC2BE79D15844C4514A505CA52324FBAD0A547C16"
01350     "40683B50EBADC87B051E6A888899945BC15311F80DB36F4E3FDDABA13112F1DB5DF6D5BD66"
01351     "C8C7C4384B86579DB85940EC9BD4037B3BD4D40D536489A6107A22E8D22F9883B423FD842D"
01352     "209FAB39616EBACA54F0056F00FCDBB2CB0906E593EF65BFE954CCABC460ED34F77B9A8D22"
01353     "F09BD3859F9439575CC08866FB87A6196C506B97FCBD1EFFB5085EA4BC0C5C540E331CA7B4"
01354     "98206604E3D1497BFC2996DC8A759A87FC6EC739B783827F8B5DC0FAA7968317B5C8D17654"
01355     "1D64C99A3274E45F80ECBEA25CFB79F0BE881D8A1705406765202F13C6E7A2F2D7CA991D3C"
01356     "85DC599F8CE853AFD50CCCB5C1A4649C29CC31A9EBE57B2C346161733C23598277151E91EE"
01357     "F57B51438E96B827B44F1CEE1AC2EA1AFAF2A011DAF21D76A125EEB85A5DE6DD6FCDF2D935"
01358     "27CA8E0B53EFE5F73855140321A4E4B23A2C7FE6B2305CA6E33F84A47DFCB13F9945BA6A82"
01359     "668217EDBC45F3958799573F7BC208799C3E0AB9684694E264BCD36CBB86EF53AA35C0FDA7"
01360     "115875AD39BF546830CF62D496968079A7EC5D618344A82D487F43E5C1485985F1C8C6A952"
01361     "8E8ECC85ED2EE473733C2253510E86A781D9EA85C7FC9B39764016334EFCBE57A599F90860"
01362     "2482E73AE0914D3362739C384EFE618C05F2899AF101CC695E57B8F55C88C0F8D3F8418423"
01363     "9EE9AF4E14758BFE5AF7B7C3FC467A47B88BB03A2E2E4B51EA7158703BE3F4E7584B67F48C"
01364     "9D0A41B5F9AE4D2BCDAC0",  // PRN 23
01365     "B9C6C5105394B0CBDC150C6A6961AE449F625D838891B683D87F05499063D51F9839480B55"
01366     "9F9998E8286A4D756738B0DB9F31B9DD2ACEE5A5DB5C67A617466FE38DA55912793390B698"
01367     "17D94E2F53BBE1254CEA7B87C49984384FD69EB341A5BE772EAF5E49929FF9E8096B190D1B"
01368     "91F9D94EC3D28AA49B4A8C78F5614BB1EB4F42E21238C23A3F574CC5F805D2438AB5DAB4A1"
01369     "ADC2DEAB9A86299E8B1D8C51478AFD1EA7ED89A450A88D1AB04D792AC98B36C62D2C803AAC"
01370     "319E0D2438DFB6495A75689D233C8A4044AA9D90AA5BCDB36A385ED8F79843C25F2855BC46"
01371     "E7F5D42CD243ECB17EE8DFB388CE95F353FE64E53E6D206C0BC40D40A179CEE7C42F586B92"
01372     "CF9E19EBEFAA8FCFBE5200FF5452BA22EEB894490BE371E200FF09E9608FE5F131BD6F1D75"
01373     "B5F9CD91884E780788B3A213827CF1C68D31C642A706C018B00B62F1877E4AB97FE53FCD36"
01374     "4735289824602004A33B392FBB67C2C2A036E1F855130A17E20B2506733451A0FA65FF344BE"
01375     "4ABDE757A7DA2EEC2FEEB00D849462A3AA88BD2F2995C475E428EFB2F0F216239425247A23"
01376     "59BCB41DB8C9AA6E9A7B3870ACBF8FCC21CEF23AA12268023F3FC83AF658E08804A7DCA8BC"
01377     "3873D0CBAD54436FA3F69AA993BB176363E08A55A346B8F7B33115BC834F4DD6CCD3E81A6F"
01378     "140B0D007CD2093585EA67670180AA76DED02CE9C55780E19411F2EB59E1B51F569CFE6009"
01379     "EFC9347BDB81BDE287DD1650EFBE39FDDBA0AB6270A4D31B3BFBFDCA6A64195798E63EB5F4"
01380     "13837996C25FB9381EA5B87928B8EED6778487FDBB84CFE82896F139FCE0705E1120A72594"
01381     "8A5841287C9E9010733142369ECF5877A41ED13989AA4E1732C64D587F2A114A7F3746BC16"
01382     "96D5EAB4CEEBAC0B022B0",  // PRN 24
01383     "A1F6FA8D82B89F75B980A5ED804DBF43B0065A591EA773674A1A76D8CA272812F10FEE1DC0"
01384     "B762250696E22D43BA240D170C1578DC85D367DA956358FE2EF23C21D526822F6070789751"
01385     "41F078D14E8C146F033FB8CDB7E6D3F8EDB59F3B6198DEFE0F32E092AAF2A30E06D90A4114"
01386     "ACD7B52F951ECE05C143D27E10BF13A446E53F8B73669FCCD7D6808261718A7DCACF96BE92"
01387     "037B8560D9E6C2F93B4F573E5697FC3C8B3FCCCB6715277D29A78265AFD2BBB3F944166849"
01388     "C925145456850E44BB206C43422DB510BCB62663E0A4D75858E095364F6AC8A7F853CCD9F3"
01389     "C26D96DB2AB17F3BE4113B3496F2C13EEFAD5F319927E245545D4773D6EFC61989F2399C9E"
01390     "66C01EF32466DEB522BAA45AE1FB784C95E2F6D96EB7A74A87619D64698538AF5485282FB9"
01391     "22D4542E563F9E8D66CDFD0F65E575F069E88202AE53945AE5AAE23A06AD33BB23F4278F83"
01392     "92BC4664AA5C11695F6F51FD0101D3163DC2E895C8CDB479750AB91416D1A3F273E7E16F02"
01393     "C712A2418EB5FD26CC9AC9FBF07FEEE065EF0D44408010CE24DC2982E7F6027067F47D2565"
01394     "27F216F18A4E93C5846948B8991282F617E18EFB8157870278C16E3528EB565F814BD79CF3"
01395     "912602FFC279CB0C7A1EDE1E9DAD9E461A1942FFA23F939D4B3764C9C3AB44E5732C9FC079"
01396     "A09CC3C0B89538D4064093384F99491115FF494F1D2DE458B2700F267FFD9FED9FDADA0979"
01397     "A20383A49945556DEC54F3AEC5413830EC09019615AA7D6BB175CA6C8AAC1D0E31C3E3D7E2"
01398     "A32D33D9FD734300768DE19F41285886C808AEAA633FE4E7B082F889AE6B46E59D6204B41C"
01399     "A0B810BF8589053D6A44E8A8139329530EC9A3963333AE7536BF6311493AD6AB1F530C4B8A"
01400     "D97D3A36E2CEFAFCC5430",  // PRN 25
01401     "DCBE11877CA625BFF2A41723A110FAD9B29444FE3641DD4E95271C2DFF6E40AC05255B7446"
01402     "1B8080C83D21BEC301F942F845677531434A814898D391646C2037E701D50B019863CEE6B4"
01403     "295AF42E574B2B78EA12824DF8CBC46805ED14DA4C741DA25653B901946FB3D95B097DC426"
01404     "8B094683B7ABF6AC6C3EF247E87146FFBCF47AD4436BD7C56EBEA788EFE96EB63B47D3BCD3"
01405     "AC9729CC3592B94FB6DCA42F35BB92C5D51BDB976D5667D6FE10F143D4C5305449D81C43E8"
01406     "4AA65654EE787B2A03F5D699D1A523A2B0BAB0D724674B8CB5B365A369A23A53F7346357A4"
01407     "077FE422B202538FC0CC0311CDCD34E104F6738FE80F962EF89CEFF0E08E1EDF89E8F6C093"
01408     "EA40494366645CE8614EED08E9D1A92B321F785C01103DAC7EC8DBC43FCC47AA9096742CDEE"
01409     "C44606429D8912CDAB0A834457FCC32759DF7954C299F2F51EA0B5F3855515C1E9776728DC"
01410     "DFE901A4F831BB45C8C66FBA13B4EC45015AB45E8370013331F26A71165FCD9E9D8C62D43A"
01411     "FEFB7669519E78510039A2A8CFF2EB9B3B53A3CAFA9818948D9A9EA205BC49638D105B993B"
01412     "23A8A544B4BAF6C5C7C72F19DE15F24E4806B4FF275C36C1AD18DD20A38E15DDC2A905642E"
01413     "26FA92C5848A0866EABDA77D6397C2BC7E7B6C416E85383C7F0102711E9599F4B0E7C06D8E"
01414     "EBE9115E450262D387930725817179D87CBD563967E3E48F7FEE39ADB8B0DA6928A4040627"
01415     "F2E96E71C726C830F0E7800671BA2C0511F757EDEDA7DDB69BB5C68817EBE9D7D4180FC172"
01416     "D816F3C889A9986B739360C99F9FB8DFCFFA6D464298E918A89CB09428E46C6EF1E9CBE619"
01417     "15AA2B0B29508C8B3004658D09DB17E6345EA20501E4FDCE27DD8F6B9025B776DB586E5C5A"
01418     "6B5D5CD64C7539535DF54",  // PRN 26
01419     "9381AD3789E337802B6E8458CC2D725126B80D82366CD6B24DB7C4C561DFF33E612721F448"
01420     "1ED6619C7A7514CA1647495B41F4E7130BF1DF6F87E0CDC1A0280B63D9791D2AE5208638E6"
01421     "1FDCF74DC916903C6E7DC3AB2F41AB27ECBE4309DBCFCF128BE89B37048D93882537A3AFEF"
01422     "92F14F22132706FC1787EE4F36AB60D14DE2071FA11817BA9B22AE462593A5CED4C0512C5E"
01423     "651E9080EC8EB439C8E48BE557C4875C9A9055FD72AF727EEAAD3774B6CB379A283EC7C0BD"
01424     "F5ED0CA5982739A3A3471015A606578718B0DE21BD323980F65F40B0A4D27DD9FE696E036C"
01425     "BDCD5350FCFB6B0D8F966AF9EE72C71329387BE2B6CEA4E9ED432F3F53DEEF22DD3E5C55BD"
01426     "C6B0CB7862CA35C2DAFB52CDF1CE7E63C502A05822E2578BDEC4CA9BE8E542CCA527120CC6"
```

```
01427     "163088F981AD109C84C7C9F190A083DBDD14CB7D4ACF8FAA5F2261780EBD89AF20870A7A40"
01428     "1CD02907708AE12D2931D457975C073D5FB241E58B4BCB6601E908A34174210469421 45D5E"
01429     "1A18D33A92114D24F405F6AAADE9ECE8C238B5F45AD7DB223C8CD1113DD3151701C99A86F9"
01430     "9BEF93B11C061F6398FFFFBB90C4385052E9ECB102AE42519E90EA7392A7FC504B6E26AC0E"
01431     "A7DEA0E8BE2EF7B85049CB6FFD7D05BF16504846D42F2C33254FD3BA377E4DF223200B5F2F"
01432     "7B01476BA96778F7F16299CF88D1ACA4F7CEACE14F533AD50EAEBA63D847662082F445CD04"
01433     "1F7D563EF2157486D3F5BBDB6F2B45620A9631C2355DBBF0535BC8FD2E9CC1E7F1E2632E31"
01434     "39585B6BB27F51C37A57731ACBCD01E87EC284E0EC1FB18ADABADA1275C4A2E45BCD325197"
01435     "7DFA9901FB524692B80AE7A0DD9B6CA604577620C71A2D0B0B110355C941111554E251658D"
01436     "460C194D5C3FD982A7FA0",  // PRN 27
01437     "E5487EFD24AD377B33607300FCD602F9FCB49730AC30D58F69A915B7912B7622E6C69A5181"
01438     "10787CEA4D3B48AB2E17AA09CB7B3F21389F977F0E0D6787AA4EA8501154B6878FAD595B7E"
01439     "A7EF437FE17901A017E3725B989A81FAC1B52736AB672D6C5BF3BBE86193F62919A033CA0E"
01440     "A4A23C51791E4AE35F96435D0DA571908BD4D63D0665B34A138E319CF65C6A65E1506178F9"
01441     "77E3E62872EAB30544152F48A4C314B374064BEFF9744EE0A6EA9FE0AAD46C5AE6ACA975B9"
01442     "DD28C66BA09F70FBE4BAA3C626CA5BE7791447E63BCA07DFD24864D0C36ECC3922761D18E0"
01443     "574F01B3EADAB4DADCF1CBA8902E405279CF31B23F21503BF58AE67781AB39D7B9C4AF1BC9"
01444     "2BD5AD7097A43DC7C0C6D7E9FD0ED15F7395B3C44E4AC78F776256A8168836C881DD0328EA"
01445     "E29DF998905518 8063B9037652C5847638E111EC0E97219442964B49453F57929061E14A9"
01446     "C7FFB0DA28706C29339B65DD907CCB39D7E3966FEA29C4E82E827019201ACDB48C1C38D5FF"
01447     "510F289BB3F2C36880509D8D25D1D3BF956610AFAF82EA89558289E5FA6B13F7A6D8479E1E"
01448     "BB0A73C0C1045FDF81C8FD9DCCC4D2D7769A20B939D5643B6BD7B5039BADC90A6DC96B0977"
01449     "23CC0B3FECFFF448CBCAF438D1385A2EB903F4CB6365898D67D4ECE9DCAC0DA810A772ADB3"
01450     "57C75357BDA4081D5AACC41CC77D9BDF16FC80B55AE9C60416218C6F2C01A0474482ACC916"
01451     "E3BF11B72C73B8E8EF6801B6375DD1FD3D5861E4E71BCC0DBAD002A488E43E7A83180AE69A"
01452     "7F7D24B1285EDFB325E819FA644C49671B8026CAC3D75255B9B068C15A83E55D0C09C2104C"
01453     "72BA1EC84DF143F5E10B9B40470FB4E57E124ED67BD3B673D6DFA785D49375201033943482"
01454     "1DADAA22CE633C472185C",  // PRN 28
01455     "AE631B48FC62F9B90929480C17F48E9E5EF285B204909B25F8933876395075BE0A02A78862"
01456     "9CF657D7117C5C8F778E93A38CD7D419C6CEE706410A33920BDBC8B3F0C0C21B27028B10F5"
01457     "A8FDB4C35F1F5795451CF52D8928658A0E692BD799A1BC06D0A231A886642483E5D3CBFD45"
01458     "98C3D4F64A1FF623B159E8EC1BA3B66549347B5DE15CB9BEF1695BD58D20B832D49145EB58"
01459     "F0141CE22477820807078EE41BBA6803428C60AB1E2B3E1368FF0630610D4604EDE0B023D6"
01460     "EFE2B4311287CB5292F4A212EF01DD0987884D12F4317F0506A87B8CFC8B6F27F452BFEBE4"
01461     "A51DD69FD0679C10633FA7F19A5558D417893BD6B39D29E19A2302A6EC4A9BE6935D7E8118"
01462     "721AA090C69C990A1C086A1E33D57833C9B561902B676641632B1325A388B76439EA0EE5F0"
01463     "1FA53D93CA61670CEFBD5CDD5D58367BD6DB99C88465FD9DD1CF5A94265EED2C6524BAE036"
01464     "76B44127CB7E4908697BF1CE93F53B2561978A3ADCAE7AB80CFBEBEB0F6E88513A9885CD83"
01465     "2AC0C3B57F9B5798F672BB60601A871EA105640C09A581C68D5525D4977F8123A275F602CA"
01466     "40D5CFD50644F09B2207D6887E74C93696CB5531435A0D364FA0649DBD0B54CB730BC31AF4"
01467     "75C28792CBF67491BD84D63754E37B1EACCDFB8EDD095E753266A5824E1AC77C02A4419049"
01468     "93405FEA4A177D3A00453456782E8ABFBC5CF47EA237BBD6DC7C5E75F64C3CEB478F250064"
01469     "36FCB0BD768674257740CC383FA717B77A3B384CD1273724356433A8BFEC603A38BD7979A8"
01470     "FBF5C66147725 0C4775A4879BFBEAA0F93FE37E77BD9AFE9351B15FA37EB5AAC87356EF4D"
01471     "3A9992E3E525D9F454B919FF892E22C13B2D93D9BF197F8C6A93375DA7F6E5630A48CEEC74"
01472     "AA783CAC2270582AA2A98",  // PRN 29
01473     "940AB0FBF5D34D53D00095DECDE97492D6C806AAD743760AF951F9DFBF68B9AC4809CD7D0E"
01474     "F7998944E5C16D60A54A12093FC3C802A5F67016430635BFC039403E244D59E6C02D5C10E3"
01475     "ABB97024D451037AA52180A23065AE2E74DF1EB04ABF4548215EFE95397B86D3C8756DD9E8"
01476     "30F96FD26B5501DC3A53CC1EBF1EB1DFB90D6D41B292A8D62AA5C8608949CF90330986136B"
01477     "9A5C0C0B2EF661DAFD21104B412C8590621C4C65D6099EE25F6E2156290EE1C679F68ADCF5"
01478     "5E37C0C0D19B785C5E63E0D6469D3A9274A7058304A05B3EF796C81E248ECD7AC85434F02F"
01479     "8E26CC46AF67832A9A715E75EEF4CCA528A07BD57B6CD51DBD47D357843915C8090273011B"
01480     "8DD8ACC9D51123E4CFD8291634E4990CA6DE35BBDD0051612DBB61C5A24C97146B61911FD4"
01481     "7E4D54D30BE406D5A89E29BA2B8772DC4EE9770ADFFE8B1526193881D5F3104355D34A9E3D"
01482     "36D322FB79032F2171F4EF2CDE53F58F76135F78BE0C2C76BC9B8E7863A4482A6F37FEEF27"
01483     "E2A61F3686909F171FFB8629F86C5EE353C9F48C6067AE66798B0C50058C8ADDE3A0E5C736"
01484     "C8AFDDDFFB67078E526E1BB31C28A5EADDC674E65963A1B57E361B60B964E3A0B07267461A"
01485     "C738E4A0FD14F07D5C5757532BC2F4699F1F9CA4541B41CC8372A6336D4AFE7052359C750C"
01486     "E0CB866CF15C01E969F8635E9C73DBD3C77778C424631882B11267663F4DD24CDBA30942C6"
01487     "57C894A1FA973E7C4091ABF858214E36FBA12CA159BC40A87630D5AE51510FF75E99E712C4"
01488     "DEAB587EA8D1D95A5767205E78923F0A6E08B2CB9C9352A773184FF2653C716A12E6BFA440"
01489     "4E4533FAEA2F40BFEB6A5FD3D34A7BDADDA5DCB008F081C45A90411F64DFDCC682FAAFFEC2"
01490     "42B90C75E1332EECF6BF8",  // PRN 30
01491     "AF07710836A5AA122AF190399AEBB0ED49CD908A821EDA0C4ED5D6615A431C42C6A9FACC9B"
01492     "33891A1C58FC4FD786CF7238B95F3E0512858F9BAF2A30C9EE0A13219A693EDC046050DF63"
01493     "8D41499E626DEDD085D9CAFD687290F19D628C29EAC839C65EE008F584780E2A2F99DA2624"
01494     "3009D599438B05874592DD1BD44E48496BF6ED2C795DD35F4E5DBB1251D63736D5528A86A3"
01495     "32E1FC959F10F99234627B62FD5F773635D448580401737 3E6C22768F16CA7B2BE2FEFACB"
01496     "F7B4853D71F8A3BC420CDD0DB67A3354DC747CDD4BD0E5BBA53E72E6EBEBEA3AF00583EAB1"
01497     "ECE8BC49876DDB3C1B51E20E954FDD7988FBBF54850B098CFD54A2730F46B80148D18EF2BB"
01498     "BFF87CE8A2CA0C61AE7DBE15917F506F58D81CB2289448324E5C85FF9FB2EC6DCDC0DD27A9"
01499     "FE8147792D9403562F171E9C400E253E10D0C004235D3926836DB7C44B57C3F9FC322E6FD8"
01500     "F22741A9FEC04F8BDE90132099346A65C72034497EC1BA5E4DB0A5E6E92DDC4AE8110A6DBF"
01501     "7D53DB059609B18D590E83DEEBB1EF8450B5FFA70D2B19A78BA53603AC17C89D39AC8DCAEA"
01502     "1DED7BDC381F5E928F425CEAB22EB54E1A388550B9B5A663F273A1592347695E2AD4E00CD4"
01503     "FF240BFEC137AB050C39F59E2359953D185863D5D9E2C9A299317797F95B77772D8CCE3B09"
01504     "E9508E6806535753B842A4FF30C6B8403F594BD5AE77FD85C6CCF25C82F453E8E5A3FC337A"
01505     "54FE01632E4E348C07A6D7C9019746A1005E3BFEAC3E4DA6D5598AAEE0F468E8DB4919CBB8"
01506     "2895779BCEA4134F13F974719614D00A17E945CAA70E64720E96A9C98A0C37531E36388796"
01507     "D27A694E8A430BBC546CE164B180BF8A8F84B675EA1B85840694479DC7B1760D89D18BE957"
01508     "E643F2D72AE6700A660C8",  // PRN 31
01509     "AE35D9E0DF692D03A1E7A1EEEDA307513BD85EBFEB24D47342C8E5DDF399DC1E9D52D36DBC"
01510     "70B22F39D20E4BADD58045171F3A9F372429B7F5162F76823FA7E04FA22CD4054B43B77DF3"
01511     "4AAA4853D49F942ABDE11CA36F9400B0ECFF00B5CF86F05C9657275A28407946D02A7576D8"
01512     "8C5273B2AF09D573866D9A1050922A66D0D8D5D9B37EFD8EAED208410F805E1C31FB76510F"
01513     "6775D9F2F4C0DB4DB9FADF52342E9D98E61E7762772E6D76B6AB2E729A4F487E1ABB49BD67"
```

```
01514     "5409BBF176C1F399B086EF42F817D99354F68CF4985A8C0ACB7B3FB70F1DE8301F46F702AC"
01515     "5A58016C61BA34DA5D02CD681327827C18D98F2C80E8BAC04399EDBCAEBE3ED7171391BF53"
01516     "16E62B972DF64F9BBFC1B58AE0B45C6ED2A59823229FA858EEA2ACC42A6F0D5029176EEA0C"
01517     "206757DBA13A723411DC2E210AF343875E36CB18211F4BD3E22F2A03F31201C2D9D3255AFF"
01518     "04DD3D6D46638EE7E772D033978E7D5EC2DE7EA1F6E0B1E15C65A7BC31B51D265E9E08F527"
01519     "E4A5C89B249200C00A1E8A84D0A0A2847DCD531FD52E44402394E2331808DC2E4F6B70A7DA"
01520     "0DA13B062833F6DD8EF3086F2B6743A41DEA63DE43D94595F7270F670FBADEE74E6036A444"
01521     "13000CF0E9CC2742964270C9F42BAFBA890D287EC5FFBC69F9271022F7E3F251B4568BE358"
01522     "4AEE97C77014E623232435F4FA98F4407535D984CCFC1F73ACAB60E06BF8870065AB22591B"
01523     "38E6B146A535B93055BB97C94C90B07E2ABA263C41A90F14B9B7DDE090DD04946886235B0A"
01524     "4DDA152EBE76444C3DCD80A9607872FDEEC65112DE6B13AA2B86230B31DB50566570115CF9"
01525     "35C5E47E5872A8E97A0873273F6090A29E30E94B239BAC1FDA20BA5BEAEC54496B0FDA2CEB"
01526     "6D99E901F169EDA8AB920",  // PRN 32
01527     "A93C92EE0E6313BB969005B1B7B56D1CB117EFA4357F8C5A0DCD7416ADC54DF5C0565DC44F"
01528     "CDDCF53BE98A2B212177BCF3A0CD287147A2C5A9A6D517A0A3C8E1373A3F5C2B1C677F0141"
01529     "2DB618F421B51EA54122673ACCDC72F8C3BC3CE4D50FFA1BE0A551F93F03AA2F46DEE94068"
01530     "84E571213A245CC4025F77214CA3149B192E1AFB80D99411590B1FF065C264C338A399B5A5"
01531     "973CBFB33EC1A8DC0223E2A3A34B02C7FF74E82708AD51386266275599913 9BB860500AB3E"
01532     "B4BDBD5F34103B23E654429E0E778E43CB2825F8FB7EAC8AF50EB6B0EA25C8EB576F658CFF"
01533     "0638F9929E02FDE8CF2CF51EB574698EF58C78B979D8F2B0F6DA53EDC935860156DB22035B"
01534     "47F307F12D39382534007E6233766872D3B61FA8748E2ECF21A207ABA8D3DC874C24609D28"
01535     "A876EBF929D459422B1F2702E1DA477AF7B1CC325546C268FD4B7DD683EA2E80728BB40E7D"
01536     "27CC1F8EC9992E3FBC51FA5740C8EE993A3F209EAD5D076DAD8575E25AA2247C886FCEB12C"
01537     "7DBEA25CE2208A23EC2EFFB4412FFB9E97530E9EDD04F52E9258DF64B2CC474B41F209D9F6"
01538     "85B79D63A78A9EFD06F1116C79C9B0D8003588BC0846C59D11BC1524DD1E2B23C17BA1D3A4"
01539     "2E673DE1D0E514CDE95536F99A067E0E00C1D15458798669A604A5900B7BC43AFF6D7361B9"
01540     "ADDB8B5E6185776148AF75A57E3946D1C0376E072C910F4466F42F8D67BB1C5F27530E2423"
01541     "4B22F8E1A96A0ECE1F2995A6741C15C91600E86F461BC64F2CCF860E524F1E271EB6D9F29B"
01542     "9C1DD54F5AC97706B72F2DDF903895E5127C581D648341F86185BE980E6D14ABAFB7AF8674"
01543     "949428553B31BE2FCC33DE92229EB21B24CD0A97D2576B58C9C80D732B56ADD6507F555E10"
01544     "07CAE0A537F82742AA898",  // PRN 33
01545     "D32684039605FF18C6881E9A5578FB91DEF05935660F2B8921F45459826EEDC6BE3CB55CB1"
01546     "489920DEE66DB5EEE7D952914C46C45FAF3BD033012D9B7F4EAB1C752FCC20A9C9872122DD"
01547     "7256821457141 22A5CD2AE6DAB9EE24CBB6FD495A7444E9D89A90721730215ECE633E627E8"
01548     "FFB23901B44DF580E08DD1912B6A1F0DFA05802E144296248A14009E92ED71091CD167E9C1"
01549     "BE1E67BAADE6D95780026F491BE86FAE681282B6895F5605559A1520FC5ADDAF49F0EFFEAF"
01550     "66BA20F0F06224FF1F8F1912BBC4A2A425DA6DB0B1DD76A7E34905FD25C3639A6465746D73"
01551     "7877B7C97433F37664B7BBB36BCEB4CA552ECF10D2F0E096454E201C88370A19A372BC7B91"
01552     "B402FE12F9EF79B479A1178263E90AFC48AFCAE32D604DBFBA15E703D8C0C62C7D0A563392"
01553     "075EA8179F62DBAC26C045E7509DB7DB70DB5CF924AF5A0A2762ABDF27A6303E312D01275D"
01554     "9CAAE9CD4963C8E06AC56C7137C1A49178C2A69E269040C23880DEB737CE0F63548994B2FC"
01555     "48C647885 33C040FC1C31971703893ED62C393CA4BA6DD8C0D5F99DD8B010D2FB6EAFCAC80"
01556     "FE292E090EDA2B440BCFCECDE414890D284AFE9D3233D7E4B50F4F6E86891F90F54A6BA768"
01557     "068F91BD5F543EDBEC049B8D825338B7B72B66054361E68FE6BF23CF3844831A9F8C855D25"
01558     "115C3D6B740B17E63703E133177F825B6703C436CD9AB2B8172A8EB50037E3B839EDD9EE5B"
01559     "DDD483377EE2933651B0A83D2759634CEE2BFEB86DD31206832F166C61458EE956C7395421"
01560     "D8B7BC646586C0767365D55E023ED453F2B38D714AB85F92256BD92FF2BF934CF375F3982E"
01561     "82A79C87F5B04B4C04E1B57E1BBA070F14722EC880FEB5A3AA4E3FC0158DE96D916F9C9763"
01562     "3BD2742DE8B60E9D8C264",  // PRN 34
01563     "8C3627FAE2FA9E2B3B7E97B0B3C62BFFEF95365AD049BCE5AE8AD9D9E57EBC427E48876632"
01564     "5D91EE1A3AF9ACBBEBD3875AAADC7561097EFE37D6DFD8B617D2A5876FCF968DBB1FFFAC4A"
01565     "F0113FE801FF722CE1093B8A4178035344E77564835D4C8C48AE00CC88FD9B9241E272EDBD"
01566     "72037C45A4A77EA7670C82B85AA2A8CB7A0B7EDB4F3B39ED7915D89E543486D75C01D8A659"
01567     "CEB6459F3E3E152656FC730D40B604A9A6C8B02D069C156CEA43224B4F9772A53AAA44B4EB"
01568     "E3E49AE222EB42151C3588178B0FAC39C9C9EBCE541D74500CDC08C8262EBF3A59DF09346E"
01569     "44ADEC62641E404C0E264C4DAA5B917AD687FE91B932F92193E5FC87649CC1D5C7C8FF3A29"
01570     "26B9255AC9167FB6BEF8DBA7795FFFCE010277B68D55554D36FA842CB2F2F1C5760934558D"
01571     "4031611EA0B5327F2345667D2B3FDD8EE06EE64E9025536D074FC134E2A5251A15F0F7485C"
01572     "767628531A02C62778BF2AF48EE3DC5092237E46430ABC9FC89F015A249427A5B935C996B2"
01573     "63E15CFE8438161DEB41B043D3D57C63B176211D3072086CCF27050A6494CB6C3FC2E72EDF"
01574     "8958B5EE5132DDCC33333C6FFDA08985E378450FECFD2442042B55E1F2AD8DE97680B2C6C1"
01575     "7FBE5261D9310CD9283AB1076BA982B563FCDCB5E922F13659DF1AB2E736D75F01C1A88B11"
01576     "84C78706473BA1CF18829E5A93E5C04FAEE6D953B1FE1434672A452921B92062F440E31E19"
01577     "441E83838CE0D3A01AF7254C8311145C07D686237574B2CA0B9CB2AFF30AE70AF2E97F10B05"
01578     "8402C23B523772703C81ACFB8D698F875F1E63B1C7D8F6DA7BD96313AC8EA9DC004D01892F"
01579     "A683D16FD9735B3E627167DD2DA2F74628555F222626A825DF2B898767968EF4C46074202E"
01580     "365153035291591EFE810",  // PRN 35
01581     "8AF5247C4E1296ED850CE9D12E5C1AE78D3CDBFF66439B5D9281FED33F3D505DFC1718AD11"
01582     "70033DFE72FA0B0FA0C54163A58E721118E058C330E76485AD282FB57E7D0BAD6C5F9E4D2D"
01583     "F4CF27835007230E3D49501C1277FE9BB9F321AD4C292E0C9E3B345E0ABB1FBBF0A81CB406"
01584     "53AF6E3CB0AD6960A10B5C30EC5B1B48B9B0D5164E1245355F7B1C54B8CBC2B333D2C6FA59"
01585     "7FAB6AA5006167781FC8B1054E284FC8ACDAEAE92D3CA520753BC6F88F50A0A516518B6182"
01586     "345E1E7DE9AA215CF4FF299AA44E6A76DCE4B4A2562B961DD118151EBFEF12E3231CF29CC2"
01587     "EFC0502FA4D70CE6626309941070BD0B91C9D947D794A304027B85AEA9E1A31542B3673289"
01588     "B112EEA962204220 45733CC5E0EB951B14FCB431B13D08C8AB3AE1D500E298D4163F2EFD34"
01589     "70131A949BD2F28EEB21572EE2CF11B54595383C25518BA48CCE91651CA41AE5C75AF4C358"
01590     "9F331EA16B7BCDDCB0A036C581741F773B9E94FF4FEE160A26CAF7107EB0C07B236B11828E"
01591     "8A509C5207CF58B34ABBB6B77D80827FB4CEEEDDC09649A740F316F9206A28C3A0A8E97112"
01592     "EEEE7C70A9A9EF04563750A68DFF06DE3D953C56677D03648612FF7BBE3BB46B8D08EDAFED"
01593     "E427AF9E8E087399E410D2F8DB33772F2371C4A9E57049F18CC2564E66E2158EA9D77B2939"
01594     "1363A86B0A2A476A6603CDC0E5D2C8E589353C893487FFC1A6C1DC22AC5560C65DF4ECC2C9"
01595     "7EADF3E87BA668BA7208253FB170D5F1B570FD027859DA05A17B739EBA433E17A3A2312C71"
01596     "529DC518856D687EEC2C245E7CBCE19F697253C9797029FB7E12079DB4715BD6B934DFE10D"
01597     "CFC083D1BE53F894570E052C1681F44B77FFC4EDB0E66CB970AB141F35D3CFA0484EE2CDF9"
01598     "5702CB2D9F58FF6325E30",  // PRN 36
01599     "FFF67C13E26F4604239AF32CA58D4EF9007ED0FAFBEF5C5653E8943FB34338096E78C87854"
01600     "F5217019C43B5B066DFEF9EA6BE0F57EC9E7CAE8CF2CEB9BE04F62506F08A54402274BFD90"
```

```
01601     "ECF1A2E66694E18CA9A1652BD72D224043B50F60D31C762852168ECF0643F817BFFC819BF7"
01602     "A9ACF8F0DE24E2F00B81E4BF75B284AC9A08836AB656BBC8E75BF2B1B30A3B278B4270DA14"
01603     "04CA821E83F3035313AC9BD8488E776C70BAC06B1089C80BCD3FB9967BC471E312CE7CA40B"
01604     "91C72EF8439F63C5A8D17EBFE9355412BE059F8C97E81D2D82B46AB626740C2E30F3DC32CA"
01605     "561118C4CACD57441514882F28E580D36FD8827BE96633868BF85AD1DC374FC2720B6AD70E"
01606     "CB0622F95A8212D2D8D5494F2509F7C152DF33466B241FD9F1934C020C4BDF29908B59801F"
01607     "8D95EDC2E87693917B042E4E8374AFD33E2472CBABEA672175112CEF5BDEBD151DD6C688AB"
01608     "6004B804C19443E0DBAEEA8AFD4AC26BB51B57F9FD08C6A38801CDA3C5E9387DC3AAD4D7DA"
01609     "7367159675733460AE1B22C73346DD412473152D08C90B7561DF94ACB36D5FC67129CA8B46"
01610     "99F4E175D88763D82D93DE0D7DCC13757C925663921A7FEAC8B6AE4B13A6CFA050CE7FBB2C"
01611     "A7E741B8C0F3E218CA356DB8E74019246526E4C7B6C819F8845528C3DCB71CE156A98E9861"
01612     "DC28E9B39AF89B0BB346B6F60A91B2867A8BF9297135101EC536E54BF05BD379EF8A3E5A8A"
01613     "523ECA0A1525FFBFE5514BBC3998B166ECB52F2E7D45B6F65D7CCF95D012E011DB513D5DE6"
01614     "1545B4E904B148EE82800D6126DD7113CC1C86570266775 83A7F041FDA18A8BFF9E4D4C0CD"
01615     "3E082F4A0925DDAC6278E533972392FADF66C25C19992DED31CFD4D216EAF001BFFEBAC525"
01616     "2C88BFB905D1284BF2314",  // PRN 37
01617     "B0643FB1F06E1D5741F2F9CD1FAAAF3C3C010014E17B6DEBA1B868357E090C186F1A7C34F4"
01618     "2AF8134D9FD4FD6BD301DD6174CC59DAC54C98F93ABB7C1B74DA21F28DFFB5715692AABDD8"
01619     "995266C3F72F52250BA6F48B0D9DD474982D7A6A3215D45E7CFC11CA95DD083F8E8FF6D26B"
01620     "B6D9D328E491CD6D73EB4E4FC98287BE31122BBD97C90AF9CAC6349A4AEB19890D5E2B09C0"
01621     "214785F523DE48CC7897B3C1878EFBD9073DDB5F06AE8ED67CD8C0702DC362C73D3634A81F"
01622     "BE7D458A8C6E71F0260BF4BA6D8E2CD8E265582BA270623212081D284C7BD6985713D80756"
01623     "FAA8B98FD3D2BA6A8F362013EC49538C45221E001EDE5E67D31F9A0B0775EC4B003CC08409"
01624     "532E405FA41BCD1B76947BFD99CDF6C7FDB07D34BCC23CD67B464A870584458C86C2C4D998"
01625     "AE7F14BB2B3764CA6228F6F3C26BF17E378A876D41EAAB0F46939BE3001B9A6DFAD5B8CE54"
01626     "DC4C986B5B1B1704BFA06843D7F7367DCB5E53C7FFE2105703F32FFF014F6975A2DD4ED209"
01627     "9F606AF1BAB0C796ECA0223F42DB8FA5C38D9FC25A8FD3027613E14272632C757A2A255847"
01628     "D1C39DE2E4236ED1A78C3144DB5DFF324537E696BF0904094CBD6C8D6A6D7E111715947429"
01629     "FF836F10D1F6E8CA45898C4AD590E13DC1B5B7BE3C419B6C418F6BECD3B130A3884DBAAA48"
01630     "B53AB54112E8E4E9F17F3448D3ACDF2C2829EC14CF68BC184884AE408743A90C92FEB9F4D0"
01631     "877931AD2BB848971C061334A1E402809E3ACB289D16C68CEF364A1A75B76E58A8E32F3F44"
01632     "480345D426A2D809D9771A6FBB5A666E5963806A91564DD62514E444861F0B81FCD65A8A59"
01633     "A8A63F98E4DFF242838CAF22A6015FCB8AD8164C1C53E7D4F39B2E8D4BED0635FA7514A2D7"
01634     "73A168D4EFA2895151608",  // PRN 38
01635     "BE8B93ED7C85573DB5EDFEAAAB80312E3C3225FF6B65457F9833D7666DA0B8180218459269"
01636     "22A42D3DBC0E19E6BF4FD81111DA2D159D6EA1CE2DEDAAD2A3DAABFFD91698AE157CD0CC55"
01637     "3ECAF1F4956E667B3D96EB9404D3590EECD2B2D600C1C2FB3E9E918EC2C90271C5B7842395"
01638     "D22DEE66B6BE16A49F698DABBDA851210DE9F22EFB2B2CCA93C6592A8D88DA6A42A4A4AB06"
01639     "2674538A1EF48EFF39749BA22DC8D40F62B670828117FC09E5537303214249199EE117FD69"
01640     "B94D462A6778B44AD2A35A03A1137D0C0023863BB92F4A587078B232C99EE4278F812A8B69"
01641     "A41F58FD13175D09C34FFF79D08FC9153DC3C896B61899F37E31F4A9C0B0EB9DABD25EA89E"
01642     "A6BB91389BAC5274300B8E27E7354CB8C3C3872CB7C714735F41F3C2FDDA43739A5C1E2550"
01643     "59E6A1E44BD6DE31B63F4BE586C3B2BC0A66F93470B159857A370A5BA305584803C0EE8E02"
01644     "C6A8C237A3A14344EEE4953AFB13DCA779C1893048E8415138C7E64921C4B8DF266019EB6F"
01645     "B3987731AF75B2F07012E91F666B982B3870F60D963A901C2872CBD029B8F0420663F65CEC"
01646     "7C573A86FADAD25BDE5BC89A21032B7F164A228B2793096ADEFD1A76209E4556D7AD8C2B41"
01647     "A06E2E1FB89D841E149D311D8AE0ABAD26927A170A785E25DE9EA3DE6FC51A21201ACB15F4"
01648     "81BB8F01DB5EC7E6F9AC61D1F58C828ED672FA2810CB3B551DF5DB3846DD1DEDA975A308B7"
01649     "BB07FD9F8F3F01D95FFBDCE2B45909F61685BF2E0C8C1D73B90ED6415300C23AFEE172DF3C"
01650     "355ABFAB0CE384929600BAF569E3C9AB65EC12C07A79A22FDB8FFDB2009E3D51C7C676A652"
01651     "7909E0AF7D9343324C94D361BF6A862C0FC2B4A4E8C7C5C7F36B2F5301C00DCC9635081186"
01652     "C2BE643ABEE415B90D1C8",  // PRN 39
01653     "EC3B51481481486D8823AB29BA36FAC0BA96AD14982DA36695859DE864C11478A630433FD7"
01654     "6A4505A2FE3DEFF8027EC93A9CCB3FBAF4A5B0EAEDB867280E7482386030068A9EA8CFA4D5"
01655     "22DF9818C782EBDFD0C5321452E4B6E883A153CFB276CB0327F141500ABC3CFF5C83C19A67"
01656     "B66701714B4C48A562F296FEC437E2431D98098CEE25F636E399B870A59D83229F4753F67D"
01657     "BB895A3E150551D0578EB4795C4AD5E027317E63AFE431B0CBF3BA94E9835CF072BF91369D"
01658     "D7414976712F226D9A005F25BF427F8EC67C8E0C82A2060137DB850722A565FC2BF73922C2"
01659     "8C687E4B914E2A30CEC484CCE9360A495FE352C21DDDB503E7DA11AA96AFC3722FB83054DE"
01660     "DDC11959AC20B7905318FA95D080E99D14CD7A6462DE93856E598FE2FB30679A1BF42F173E"
01661     "08DDF79541B1932B5AD345C974BD025EE5AE9AA62F77C0C2D6DB056FFCB5B5BB91028DB44B"
01662     "653C8EF3D684E42C81E2B2D84A406A3ECEC4665737244D390937AD551AACBAB4FA3EADF47F"
01663     "7E39E30F0656430B5219B6078B095468323187C4A9176708A240278869EDBD54BECAFA13BB"
01664     "FD86367B7248B567A27A78112E13566FB4CFAC539F364F9EE30C130E7547BF48E939DC61EC"
01665     "D4900AEE4D86F191368501D77D06BC327F5AA76109E42D18A6447D2CEAB57A065417E2FD20"
01666     "D5DFC6A6F84937F2514CBC37EEF4FD5E679E451F8862363E40AF7620184CCBD55916E26CB7"
01667     "3B71E674A1E6502F6D8CA74894933C444EEEC1476526AD2CD7CA9AA5D918D7F2EF23E4BBA1"
01668     "AFECF9959C801C3C3F3793140F69CE250FA9B954350AEC14EC47D47E7EEDE377223FEF7052"
01669     "F66EB205598BBF8163673DE410DBC817459930328D63C35FAF57C1010BEE90CB6D0FD80C13"
01670     "60D48D20D6143C705EC34",  // PRN 40
01671     "BA8137844CEB25E531F33EEC1F8DA7589D85F2EA0B99F51040EA36A9C6FA16FF993E676A2A"
01672     "994EE553CDE801E138DFDA6A797D4E592F8647606341ECF17D8746174DF4668FD108317D89"
01673     "DCDEEBC7A86B0FEAA48AFF0DB1E2A4D02C4C8145FB259D8411EEEBC6F3A96B85869FDAFA0B"
01674     "5C2197E1DAFE3CF16EF6A56B057CD5463011B03BB810E26AF4355DB962F510BC71F8602D8A"
01675     "9DDD4244772DE59C638FC3658C64C600CB69EF3BCA2904C242427DF431D9F47488FF324F38"
01676     "E85BEFA5B66A977FD9809E32A9EEF99A944612F1D65DE6794782E278CC8AE9EAC377C5A114"
01677     "B0150678D40A7DC7ADE021F289A657210AC0C84B5970D56A3629B6BA7B3EA98CC9EB575FD6"
01678     "FA210DEC335E82B1963FCF0DE32DE0AE12884FEFFADF33F7AF874324A712824A37E147F7BE"
01679     "C3C2238C0C45C562A6D48AC3756FC21D62BFDE47AF915F51A34202A994E7222954CBB1EC3E"
01680     "E7F6D691A53BB2E692CD80BEF07BEDF47B02C2826F1373AC50E317BB57183A935121B913D6"
01681     "A8DA1AA66A749FB25EC08CEA10932F49077A258EF3C9B24DD8B265BF14E4B1BF072CCB833"
01682     "FECDB250B400F1D1B6F4BDBE460CAC0BD0009585BDF11609EC8AD8C75A341B5BEE2A55776D"
01683     "18DDA2AED5CE666AD03386BC86F0369E529203BAD8940F5A96E98198D94BF6B9E6A48F5B7D"
01684     "3ADA260A19336590BF98861722098A4CDF4CBF35E8C55DC0D8D0465066AF954654E76AEB69"
01685     "FE0E634DAE60F972841F79AA68604167C178FA11127F30994D667B1169A4B1A0649BD0E13B"
01686     "9541E3E1F530ADB22DF21C92DC17F7CD289F5CC24EAD69503425AAD76CC14DF302993AA842"
01687     "D40121491EC8BC040427D7DB30644881CBE1F3BDB93444B46F42C001C1144D265379B6085F"
```

```
01688     "C1C73ACF04A81A1302F20",  // PRN 41
01689     "8C296D08BA204B2B2C7E11D7E6959FCAFD027202B6D92C4092940D3FB268653AD99406B5FD"
01690     "1C92DCC3BB11D44F38F23F2B88052C1E48E34FDEBADF8C18D851791EF78EFDC2A34BDE6810"
01691     "9410C00E0146804C2C5571261C9483B6887B7769C895B76FDD162A4A99FDEBB9DCD6A04CD7"
01692     "A6328E7694E3C2080A611665FA2508F64332E5D8AB3357E36753CABF60725FF30C6F6E0E82"
01693     "E80DF32F6DF9BC11EA9CE6F0851D1B0CBAF0732AE05A307BBB39A56A28B164D6459EC3AEF0"
01694     "52D517BFE62E9798FAB52BCE087E42D0D8DC45625A6F7A32A7E67305C8000E5738D2C8B53E"
01695     "82F10459665AAF2592CDA2D00792532FB24A77802482B4C70B38072E738457A80CE8D7E2BF"
01696     "D049E4353F929FCE6E6AF3091695080CE93C11C8B3C177C51BB9640519A81EA7A4A1CEDC26"
01697     "D128575A281DFA3499E5146E0D47CDFA9E8885E168AF35C64501B9AE45DE4A28973098FE8B"
01698     "6E809D357503CAA5BD373F767B41763F6864F42E8B7B7EE40D88A137AEF2D22B5F95FD973E"
01699     "EE370FB3CD9338E15547B5A1934D50110CBF3E4B8DC070B323CCF905CFCAA6DAA359739AFE"
01700     "B7C2B0CC98996D202FB51ACDC8379709423BB7683294A2D7AF0DF6B44B8243DA21DC829630"
01701     "5F89ED52F0EABBCD1DDF19044FC033B5EAA5BCA6FB93EB7ECE4B8934C3A35D0F5380C5B4FE"
01702     "526CDBA2F473F233A4F32C4C2B4A3F9A47F96FBF0023CB8FF2FA1CCB07717CA449AAB5BD98"
01703     "9B6CAA8D52E046AED4BFB1B2600D3F3BD037F096F006389C3BD875B6064CFBF44C2DE09962"
01704     "D1F30995D32CA0F5474F41BE202AD67AFC1049870FF330C6887E3F765CFA3A85ECF11182C8"
01705     "67598AA61C6997158A367D94E202709D839960E6311AD687ADAA05A5ADE0088E6B492B0F7F"
01706     "E053A67CB4DA4F3CEDA88",  // PRN 42
01707     "E40E6EF4455C67D6EF9D6A72858B9E6E08AE3B48EB5F0339A02BACBF979E8266AD92F3D100"
01708     "2DDE744C69B7C92656B3239925CB7910A50EB4D03DCD28908ED3D7F560AEA47AC4B0EC9D7B"
01709     "91AD12DC15920032C25CA1247FC7F69B148166B6E61629C3B84687B9045AD65FE45CFBED41"
01710     "34B68A2CEF9BE972482C0A9442ED0A2D48A55F2DCE46412BE557D793C26C98A7F68CBE30F0"
01711     "70435F552811CAF34F2C94EA0A46AC4FA41252D1AB745B5647AEC23A70EECFCFC0B88F9313"
01712     "8C45140CF23AC030C6571BE79520684B0785CACA1D918ABFAF2FA925D0E194C85BC7D750A1"
01713     "EC4E85E51111ECE870361DBE27791A3A2E26C80A7C0B1AAFB0C013C29824E1708D7DEED9EB"
01714     "4C55BB5DF732F361BB6F3322166854F8ED0910E29FCC51B86549D47CFE2D4B7EB110B306EF"
01715     "FC076F37FBE276329C70DEB6308D9B959447D8604ABE6F18F653E141142E0C77F08744CD49"
01716     "0C284D2C41410786F892A662821BFA5CB5F0FF00ED7584B576E8D3E0D94C0D1A8B1B532A64"
01717     "F78BFC8AD8E676799E8970349828A780B76A76DE7625E6C76EABE4AF12D7F80E3E43AF3DC2"
01718     "D7EC10A7AF4EC6C3C158AE63D82FB47EE6D98F9043EA025448254A34F4FDAC70C139B73088"
01719     "026AD9607A3B04FEC00BC9E6B86E30C4A606D470FB0E022200A786133CDB870BADEBBF8812"
01720     "81C8C02FD8DAEBC229DAC57C9F5B49E75551488968821DA88FB7778F796030CA04E78E53BA"
01721     "EE835D2D51AC62564E579291F205AE7337774677394A326D1D7C1E1E692B7738ABABD3BBFD"
01722     "B1F0E999098FDF74B3817A19E6BA5CC4A41F6760F2A57CEB0C3F668BA49E0E52065B3C3072"
01723     "34D5735EAE7FE7EF5ABA539C2178A321ED5A59E7BE313757C51969AF57486FBEF7757C2558"
01724     "A1599C996C8CB823A580C",  // PRN 43
01725     "8B831E082C933E273C83032428539C331C131145AF171D3CB5C4E46C44E2CA0E185EA893B0"
01726     "69CABC0DBE995C6A123ACF7D546CF4F0FD15FB0F862B4983845F5CCE43F01EA0B2D6523A8D"
01727     "01BBEE1956EEDA8102AEFFD2EDAEC51644548744D6B51D76F03EEF031C4B5486CED1C94B06"
01728     "9F1F73E53B2027D028CCF042FA100CDB6BBCD82A4D9DF60A4E7C70664AD2069E57365B0F28"
01729     "C793D5B8D3313AD3E582404918F0258FD7D70FE878CE8B4DD4DBD092BC17622C46EDFE8852"
01730     "B638AD2B60AA4784DCAAA7CA2F40069D675AACD131FEB0A2E967769670E66A3825ABC8BA6F"
01731     "5ACC7EBC505CD2BDBE148F21B0398237076CC593D9BCC457BE7A522E3C5E1B0B6A6EB5A4D4"
01732     "CF788DA7C640FEEA1B28E555B21FB0DAC6503F70D0137387BAD7AA915C9BC7EA692573FA01"
01733     "2E4A5FF41215F4244AADC04342917B6AE6BA63B4A46C0B715FAC82CEEDB6228BDA51B4548D"
01734     "C477EECC3FCE6EECED8B15A458420CE6BA48798D7FAB3295C9AE161D7FA1DED73858A29D69"
01735     "8C1D4005A75D914659603824C5BA2FB807EB63E60DA4C47D3A43D59FAB931E9BC50FEFFBCC"
01736     "AE39307780188 49C388B405EE902183A69A8FE2645378C0144B05CB45BF88FFCB9FD1D19C0"
01737     "283EDC6116C66D5A54F6542921D4B423A3CBF2530EF534CE01B735F49D5BCB0634066B0617"
01738     "EC916E7076CF084B30BA248730645D299C37B8B6B181C67AE492AFAFC33241D22336432DEC"
01739     "1B6F74194CA5821417AE9D4AEE87FD30E77800DE6697FD1040F9BCCA86C2B838F3973591C6"
01740     "3F843D875B3A2023EAF38DF399DE1DFB696C8D06EE8D75FEF76956AFB612D5F4CF06DB484E"
01741     "E4E31D7B3C4B5D371EE92EAD3DC6C1D0067D57B822C9F70E3CA17C065546D678D04828BEB0"
01742     "76572B5582FC6939EC5B0",  // PRN 44
01743     "96FF996AF5BF7DBF540BC4737B5F5C0C6CABAFDD3A821A2DA1F597066FC4DB438D8918310B"
01744     "3184EAEE80F2F383C0B0619A1F1F2F77C7F4DD055363C3795EDB3036F582E813D7F44518E8"
01745     "E959839ABFFFFF2DE9C657C63B9DEF44D18474BE37AA6FCE42A0D2C8F1BD4DE0A7B674A44B"
01746     "4AF2E67AD12411B0F6113AC6084B5B1F5260E973CF98C0E96B3FA80A224C0BB3295446D538"
01747     "4FD94A3BEEAE7CF8AA7A71313550554AB13E1179D6BAF847D0E3D898E66683973871F989A4"
01748     "0D3353E1DB83EAECA00345DDC6D5D2987D666624B38B9EB0921253B5CFFC2DC9FA5E226F26"
01749     "59EC8CD131BF7E4093C81F99EEBFFE27B40319E2D4052502F56E79042EFB042709FD035AD5"
01750     "8203DE6180757C5B7B0C471AF3612E93EA2442B577039AD52C68E08DD9B377D96ED0A453DC"
01751     "CB5650392DC2710EF24B2BE9E3F3D0CEE96F7EFB18ACC226F022C136FC0FDA126A0E1CF7EB"
01752     "17C1A8AD9E9110B24A4A188A51C19FB67A514F80A967F847C7A4E4BC1BB0505D9865FEC26F"
01753     "D91813D10A263421503856AEE4DC1C6CF50544EFF6C685174CF644B729C28A4AAC12B44876"
01754     "B09C10FC6766313A226AD1CB2881903AC4043DD36E8BC1B302653B0035837AC6D1D349A92E"
01755     "577F4E84ACDB2532BC954CB08ACFFBBDB1D28AE65951BC79F78CC755925A565268BE2441B1"
01756     "D40DA1F450B1EFDD2BCF8F7815D8DD1F3B05D8F98A838944B60DE76DCE52C36FE5DDB4F837"
01757     "14985357831A536A95EC0BAAA9A94C96F86B342DA911FEFBA959DE0E58999DC13C16EB9792"
01758     "B5B92288FF16148F910A5B06D8D05457B12E15619C025B13E4D93472E66F80AB49A23A2201"
01759     "FB2817A61273291500D84147BAFC8A3B96BEF241A7B0064F13AD2E0C84C82BC1F3CFCEB8A9"
01760     "3B927D1B58939AA6EC018",  // PRN 45
01761     "87FC908811EE777B2C509E835CDA0260B65E061E9E508F09B49DBF046446AA33C336FDF490"
01762     "573B177B6C8C791B3B1CC26A874FD9BF1AC54E0D3841A04AE293B5D9E079EBD219082C402F"
01763     "56ABDBA94930E91F99273811E8198E31EBE31E466AAF25F002BA7A3E09C7FC4A5188FE5616"
01764     "BC1E1A5E54B663B502AD59B67579072DC93F49CBAB351B9E563BD67F56179D8370579A8FC1"
01765     "4133C4E9437434A5E1D0E53E398EADD2C0BF0EF6542EF6101944F25B96EADD0B3853D1D27A"
01766     "C64108701E3DCBB0BF7F45B9C173CD4207E267090CF7870427BE248713E064111959D2D27D"
01767     "B5510376AAE140C5FB79D4AC873E55A3863A0FD183E4FC2D1B5D6A5B942F74A79BB64981F7"
01768     "189FCDAD06793891B7B97BBA3ABB8BC88976D8902B56CA9D548182C65F33D3C3121A80817C"
01769     "D1DF6B4EBB07F2F40D03443A7EFDA50812ED4E4AC241EB86596146D3228FE464D849C51CF9"
01770     "CEDCB851AFFB2C92DB16CEF971B6A352BD1589289C0BD619645A0140879923A9588AFA38DC"
01771     "097DB0C006D82C0D4E1B6B273E184EC81BD688E832B61D39AADD13C2E27E0D57CA72C1FABA"
01772     "81CA38936473BC3C4C79C6635D265172A26F794A2AA8175DCB91792871E14258717AFEF66E"
01773     "DCC1AF2E2D3A232DEE44A3CFB24696324E466DCF10575C0F14562DA1AF866ADF26A4D54EAA"
01774     "888F231327F006716BF1F4F801E09FF7FD4528D2B4FC9247358EA7B9AA2EA2B9E80F5660CF"
```

```
01775     "D19B77A9D83379493A0BBA1FEF3B998AA45DBAA7D5E6F4BDABD00C86097576E199A9C0E8D9"
01776     "E91794E3D545AFDA0AB92A62B66429F3668762B10DBF683F77DD2FF99F92B94B3060D8747C"
01777     "34C581102E21C6596248AA139283377A3ED85DA3C999A6FB029780A9CA1DAED1C9B889EF1A"
01778     "1441219E3AFED76F1BEA0",  // PRN 46
01779     "ACE6BD607FADA0C25C3F74EF23BAF6E29C70214318035E80B160E5485BD57C691605DE7783"
01780     "B9EFCF498CE5B4D8EE6619191DA2A15820015DF7F0B8AB05B22562DD437FBE3AD141AE0E6A"
01781     "E45313FF915A39A49466F11B651CF210143D6FB06BA6E5BF4083671FB3DC9454E4EA38ED98"
01782     "0C5C2EB65F24FAD970316D5BBF490B2403BFDCA348A7DB1B4AFD9A91B6898381CEEBAA5F6E"
01783     "A1419C19D21E1A64DA2063D824AADF7C0E546D608F7FF19B89BDD9E8808A7A236E98DF0728"
01784     "BA39498DC6BCFE82C534DDB6FE02B303B2DB401DF41C53F4F5321CC28861D22010DF2D1457"
01785     "F9C594F3A3DE00753CAF0CB6EE6B34DA2364EDDCC2619930C7DCE4C976F50BE69444BB2759"
01786     "44D17D07061A87D1CFE0DED63400B514C6457F4119845A19072678017FD6A498CE3D3932CA"
01787     "C6B63335D87426F1C16FCE0F5861FF0953CA13E9C911A947A2C82231E61012DB7E782BB839"
01788     "764EE42A94FCC7BCE4BED84D891EE959D9430B963CDD38EC2F63884AA493BE7FEAF47B6252"
01789     "F2F042531A5FF5347AE9C322C374AEF6A7F4529BCBD085696178D1D2C12C9AB91A2B66E3F0"
01790     "5A45A1950BBFA797134B4430DC09289B09EB1BD88D05B03D2EC269500532FCEBF7EC7A652E"
01791     "A9F96AFB622E6D71E568756C2D1700D55238AD36D86A6D92C62C4522F4EB2F2AA0F9FAB8E5"
01792     "6D4052B13890FD0A9BC17376E248EC86AE21EA8430B110A8A2AA835D7317095CCF2FC00ABD"
01793     "132A1D2CB7CCB105871E4FCEF7F5D8B7CB3E9FE24303DAF57EE80581725B9ADCD9CE09F7CA"
01794     "50DB4B2B7CF8AFCF7C39958F4A963E92E3E29F4349275C772F68D135AE05CA99019C14166E"
01795     "D34E0C3598A67C9CD8CA68832D7D7EEECA0BE86B4355B8439002D2F7CC7933F07E6A8F0908"
01796     "CF0F3903996C390142728",  // PRN 47
01797     "9F4DE194AFEB87B4D4934D86F08C2F3506BFC8F0F0AE05E9AB45A17B36AA9C7D5534127B3C"
01798     "98E60885AC2D00006A8CBA4EAF711CE0C9497B04B539553ED80B24EAC9496BE53725F8EAEC"
01799     "5DB5BDB71664C45D01CD17BE31D039CE8CA16FD9D9D60E9294C8ECFF4975392793C879BB2E"
01800     "375AC7EFA47B32D6D895E6A3F929E78BE82EFC896E89304FC3F3F0751E22D523D3117AF239"
01801     "B897B12841429AFDBE8F8D303D50D2A56275CEADAC04CC32A95E9DDB45F4D4C83EA7CA29EF"
01802     "5E8F7D172D18134E5FE9BC9E442F6D2DD3B01807B994C80697A28CAC0927FE31E62985F629"
01803     "A965FD14083E72D65835BF6230A660F10D91A04EA9809FF8817129E6503F28596FE9A486D4"
01804     "86A3B789580E36C0F9C6B75BB6548742676F50FDA64602EE6986EF10824C90E66585A118F5"
01805     "C3BD1FF67BBCAD408CF0762D59216884525FCB32863B6A1502302C4F09B3BCB27A9EBA72EA"
01806     "BE68D86A15CA6FDCCE89CD4F20055C3221611F1B1F9CFAD8B93A68C42AB19565DABC486B0C"
01807     "A93215DDC8E128806B4ED06FAA0987A49EFCFC1C23E1BACCB6ACB332D720311B8FED658024"
01808     "F0CA12EAC658494DA79F9CB56D59F17C927650575562095A99F5A11D13CD1267BE89FBE761"
01809     "DF5CF0E9D8C67A9DE21E8BEA108BC3E2E3AA68C7742103B6F975DC232443C34F17D9EAA193"
01810     "9C23514BFF908757849AC3EC6E954FB4991373DF8E7DC6011D16F9C19E4C7021A788C0834D"
01811     "6BC09A560908AF50F71D0496A3DE1EA1DF8F485A3C496B9F4B65A279083F8390966C034829"
01812     "6D05BCFA81D2EF8A2714C4A886D35DF870BA4DCF73BFD8E3EA1DFD4D9E37A0ED8E9032E350"
01813     "947D61B0CA2FE10FC20B8A5AD0E3FD70AE3C8BB91F2E04619ADB3C684E10B9127E5B76B8CD"
01814     "0E9BAD2FD100131E6FCB8",  // PRN 48
01815     "953E31E5F9C45E57DB6C54DBEBE650C07F8C14D832BC39D5D860912A7B4EE9F726C0D3FF84"
01816     "2DA45E3E9286321AEB6A45FAB260E1FB3AFB7A4DCBFF3D65491BC61AE83207E1CC56858D13"
01817     "046191A5CBAB51C59F3BB37E281847EE624A8E54DA17273262E790246A54C8873BE252E9FB"
01818     "D7A7ECCD476D62124FD1F28AFE207545525B325DC6F73F25D09E3589BC848E0A803EB10FF5"
01819     "0E81732E5F317CD3AADAFA6C123807677B273B6B56955065869078BBE27AA04CEB7ECCA493"
01820     "DC2E9499188C51B16309F014624554EB96E3A69E3E067FA946977379B4BD23A9DD57D5E175"
01821     "FAA891F5FCD72431A3C857A582B233403B0D13C48C502F9E5324DB127DA83F019272A29ACD"
01822     "F325E6AB2908ABA54CD56D1D91188B320A1941720BDCC2CF00991C932FC5D7025B69F242F7"
01823     "75A06DD6BD8FFB7615246AC4B7495A350C4BBE9F65F7449969EDCE013BF3F7367647316C2A"
01824     "A55F882BB42C7BD7C43618F78A6409A8659919E3FA20DAA9BC384CC6BEA776A438080DA6DF"
01825     "11AB8A853F06DAAE429883F3C153FEDFDFF9D968FF806D9B7CA193F3A72D316E0297A01CF1"
01826     "AB6350D2E807F93EC27E7BD91CEB6CDC44001E092EADF5523BF0D67EFDCD89D8E5A2F7ECAF"
01827     "0B126F4D2AA32A484B5272AD7CE9E9F26CF418E66429271244830433E3C15B7833655C6388A"
01828     "26A10152857221718FB9AFE36331E071C04E498DA3181B5F5BC701CC5C30BB963183F6AC5F"
01829     "26A5CACB32ECB7C8A88FDFCE73823FA6FBD965B2772067CC2B81BE5A35660ACD3F5658ED28"
01830     "A0D225B54392F0D03983148ED738FF3A80E5DEB5D2DF3092C1AAF81272E75190397F9FA66A"
01831     "D3C101F259496F9EA011082D7BE7E1E10F80ECC4C42AF40E0AB498081404E5593A71147038"
01832     "3D8149ACE622AA769A900",  // PRN 49
01833     "CE11D9F5C85027678446809BB43074F041568F30717E5B784BD0B64FAAE0610A144496CF4A"
01834     "AC1BEE53CF98AAB2BA75239D228BF946D99CD0899C250CF1850D17ADD418E389F4ACE5A942"
01835     "6A1251074E28C13A818477875BC0FF732517A6CFB18A23F633E9D0F876582907AC8C3403A5"
01836     "7BBAC9706D7BFAABEDF132A3990E6C4089BAD9939F1F8CE910779BE5CF96EA87716BDC45CB"
01837     "5C7BCB5D14123D03BE2E50909C523D10096335770D849D509811EC426D80E1115B3AF367D9"
01838     "41C74D8025D7C7F5B2AEC5E5E7AD5F38538CFE33387B5D19AB846A7CFA6C6CC4E0AFC054DF"
01839     "EDCBDB0A8B489A082B3B2A064EBE0902AF6BDA5B64310641F2D717D06E1064D2C2C0ABB19C"
01840     "66DFBA807E36ACE0F6BB487C9868C2F354909960BDF7A8B4C42CC935EDFB10C1975F051D40"
01841     "29A76D255A1CBAD199393A8A7BBCF80C25649C22B475244323D5323675D74886B30A90B9B9"
01842     "1939521DE115DD5AD0509363C6057B9DB321C0871D45B9269B048A98148BD91E3976DDB740"
01843     "5F0AA990A8C7987767F76AD1CBAA8B442DF151E28FFE3D4A985BFF6C263108D8873CE28F62"
01844     "DEDF610ECB1A6C35FE5D66F437B227F184D856EF994099763510425B16234A31B14FC8BD6B"
01845     "C37264F9BAEBF838AC2D03BEA98EA7F9FA0160A21BC211EBFB82A5CC2A70F72FFBED8170AE"
01846     "C8209A6B7ED8CFA7A6FFB9E956239C9A50249D14E365EA7166033BC83D1FB6F336A8BC85E0"
01847     "EB6EE26EFD4B8A5028CA26F9CE872C7A3CFD21CACF29E42DE0309211CCC8143C7CB336A450"
01848     "A2B786D526F53265B60339ABC9ECA67FAAFBD3CE1DE3C3867A9D11D758AE3162BC3D464EDD"
01849     "1C77EA1A4B4A3A0DBD80D05E566F97DF2555E427C8466AE006876402B89FBF37CA38ADDBC2"
01850     "083D30D7F27DAEADEF334"  // PRN 50
01851 };
01852
01853
01854 constexpr size_t GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH = 25;
01855 constexpr char GALILEO_E6_C_SECONDARY_CODE[GALILEO_E6_NUMBER_OF_CODES][26] = {
01856     "83F6F69D8F6E15411FB8C9B1C",  // PRN 01
01857     "66558BD3CE0C7792E83350525",  // PRN 02
01858     "59A025A9C1AF0651B779A8381",  // PRN 03
01859     "D3A32640782F7B18E4DF754B7",  // PRN 04
01860     "B91FCAD7760C218FA59348A93",  // PRN 05
01861     "BAC77E933A779140F094FBF98",  // PRN 06
```

```
01862     "537785DE280927C6B58BA6776",  // PRN 07
01863     "EFCAB4B65F38531ECA22257E2",  // PRN 08
01864     "79F8CAE838475EA5584BEFC9B",  // PRN 09
01865     "CA5170FEA3A810EC606B66494",  // PRN 10
01866     "1FC32410652A2C49BD845E567",  // PRN 11
01867     "FE0A9A7AFDAC44E42CB95D261",  // PRN 12
01868     "B03062DC2B71995D5AD8B7DBE",  // PRN 13
01869     "F6C398993F598E2DF4235D3D5",  // PRN 14
01870     "1BB2FB8B5BF24395C2EF3C5A1",  // PRN 15
01871     "2F920687D238CC7046EF6AFC9",  // PRN 16
01872     "34163886FC4ED7F2A92EFDBB8",  // PRN 17
01873     "66A872CE47833FB2DFD5625AD",  // PRN 18
01874     "99D5A70162C920A4BB9DE1CA8",  // PRN 19
01875     "81D71BD6E069A7ACCBEDC66CA",  // PRN 20
01876     "A654524074A9E6780DB9D3EC6",  // PRN 21
01877     "C3396A101BEDAF623CFC5BB37",  // PRN 22
01878     "C3D4AB211DF36F2111F2141CD",  // PRN 23
01879     "3DFF25EAE761739265AF145C1",  // PRN 24
01880     "994909E0757D70CDE389102B5",  // PRN 25
01881     "B938535522D119F40C25FDAEC",  // PRN 26
01882     "C71AB549C0491537026B390B7",  // PRN 27
01883     "0CDB8C9E7B53F55F5B0A0597B",  // PRN 28
01884     "61C5FA252F1AF81144766494F",  // PRN 29
01885     "626027778FD3C6BB4BAA7A59D",  // PRN 30
01886     "E745412FF53DEBD03F1C9A633",  // PRN 31
01887     "3592AC083F3175FA724639098",  // PRN 32
01888     "52284D941C3DCAF2721DDB1FD",  // PRN 33
01889     "73B3D8F0AD55DF4FE814ED890",  // PRN 34
01890     "94BF16C83BD7462F6498E0282",  // PRN 35
01891     "A8C3DE1AC668089B0B45B3579",  // PRN 36
01892     "E23FFC2DD2C14388AD8D6BEC8",  // PRN 37
01893     "F2AC871CDF89DDC06B5960D2B",  // PRN 38
01894     "06191EC1F622A77A526868BA1",  // PRN 39
01895     "22D6E2A768E5F35FFC8E01796",  // PRN 40
01896     "25310A06675EB271F2A09EA1D",  // PRN 41
01897     "9F7993C621D4BEC81A0535703",  // PRN 42
01898     "D62999EACF1C99083C0B4A417",  // PRN 43
01899     "F665A7EA441BAA4EA0D01078C",  // PRN 44
01900     "46F3D3043F24CDEABD6F79543",  // PRN 45
01901     "E2E3E8254616BD96CEFCA651A",  // PRN 46
01902     "E548231A82F9A01A19DB5E1B2",  // PRN 47
01903     "265C7F90A16F49EDE2AA706C8",  // PRN 48
01904     "364A3A9EB0F0481DA0199D7EA",  // PRN 49
01905     "9810A7A898961263A0F749F56"   // PRN 50
01906 };
01907
01908
01909 /** \} */
01910 /** \} */
01911 #endif  // GNSS_SDR_GALILEO_E6_H
```

13.702 galileo_ephemeris.h File Reference

Interface of a Galileo EPHEMERIS storage.

```
#include "gnss_ephemeris.h"
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <cstdint>
```

Classes

- class Galileo_Ephemeris

Enumerations

- enum class Galileo_Nav_Message_Type : uint8_t { **Unknown** = 0 , **INAV** = 1 , **FNAV** = 2 }

  This class is a storage and orbital model functions for the Galileo SV ephemeris data as described in Galileo ICD paragraph 5.1.1.
- enum class **Galileo_Nav_Message_Source** : uint8_t { **Unknown** = 0 , **E1B** = 1 , **E5a** = 2 , **E5b** = 3 }

13.702.1 Detailed Description

Interface of a Galileo EPHEMERIS storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es,

Mara Branzanti 2013. mara.branzanti(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_ephemeris.h.

13.703 galileo_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_ephemeris.h
00003  * \brief  Interface of a Galileo EPHEMERIS storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es,
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GALILEO_EPHEMERIS_H
00020 #define GNSS_SDR_GALILEO_EPHEMERIS_H
00021
00022 #include "gnss_ephemeris.h"
00023 #include <boost/serialization/nvp.hpp>
00024 #include <boost/serialization/version.hpp>
00025 #include <cstdint>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 /*!
00034  * \brief This class is a storage and orbital model functions for the Galileo SV
00035  * ephemeris data as described in Galileo ICD paragraph 5.1.1
00036  *
00037  *  (See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf )
00038  *
00039  */
00040 enum class Galileo_Nav_Message_Type : uint8_t
00041 {
00042     Unknown = 0,
00043     INAV = 1,
00044     FNAV = 2
00045 };
00046
00047
00048 enum class Galileo_Nav_Message_Source : uint8_t
00049 {
00050     Unknown = 0,
00051     E1B = 1,
00052     E5a = 2,
00053     E5b = 3
00054 };
00055
00056
00057 class Galileo_Ephemeris : public Gnss_Ephemeris
00058 {
00059 public:
00060     Galileo_Ephemeris()
00061     {
00062         this->System = 'E';
00063     }
00064
00065     double Galileo_System_Time(double week_number, double TOW);  //!< Galileo System Time (GST), ICD
      paragraph 5.1.2
00066
00067     int32_t IOD_ephemeris{};
00068     int32_t IOD_nav{};
00069
```

```
00070     // SV status
00071     int32_t SISA{-1};     //!< Signal in space accuracy index (-1 if unavailable in the source data)
00072     int32_t E5a_HS{};     //!< E5a Signal Health Status
00073     int32_t E5b_HS{};     //!< E5b Signal Health Status
00074     int32_t E1B_HS{};     //!< E1B Signal Health Status
00075     bool E5a_DVS{};       //!< E5a Data Validity Status
00076     bool E5b_DVS{};       //!< E5b Data Validity Status
00077     bool E1B_DVS{};       //!< E1B Data Validity Status
00078     double BGD_E1E5a{};   //!< E1-E5a Broadcast Group Delay [s]
00079     double BGD_E1E5b{};   //!< E1-E5b Broadcast Group Delay [s]
00080     //! Navigation message carrying this record
00081     Galileo_Nav_Message_Type nav_message_type{Galileo_Nav_Message_Type::Unknown};
00082     //! Signal from which this navigation record was decoded
00083     Galileo_Nav_Message_Source nav_message_source{Galileo_Nav_Message_Source::Unknown};
00084
00085     bool flag_all_ephemeris{};
00086
00087     template <class Archive>
00088
00089     /*!
00090      * \brief Serialize is a boost standard method to be called by the boost XML
00091      * serialization. Here is used to save the ephemeris data on disk file.
00092      */
00093     inline void serialize(Archive& archive, const uint32_t version)
00094     {
00095         if (version)
00096             {
00097             };
00098
00099         archive& BOOST_SERIALIZATION_NVP(PRN);
00100         archive& BOOST_SERIALIZATION_NVP(M_0);
00101         archive& BOOST_SERIALIZATION_NVP(delta_n);
00102         archive& BOOST_SERIALIZATION_NVP(ecc);
00103         archive& BOOST_SERIALIZATION_NVP(sqrtA);
00104         archive& BOOST_SERIALIZATION_NVP(OMEGA_0);
00105         archive& BOOST_SERIALIZATION_NVP(i_0);
00106         archive& BOOST_SERIALIZATION_NVP(omega);
00107         archive& BOOST_SERIALIZATION_NVP(OMEGAdot);
00108         archive& BOOST_SERIALIZATION_NVP(idot);
00109         archive& BOOST_SERIALIZATION_NVP(Cuc);
00110         archive& BOOST_SERIALIZATION_NVP(Cus);
00111         archive& BOOST_SERIALIZATION_NVP(Crc);
00112         archive& BOOST_SERIALIZATION_NVP(Crs);
00113         archive& BOOST_SERIALIZATION_NVP(Cic);
00114         archive& BOOST_SERIALIZATION_NVP(Cis);
00115         archive& BOOST_SERIALIZATION_NVP(toe);
00116         archive& BOOST_SERIALIZATION_NVP(toc);
00117         archive& BOOST_SERIALIZATION_NVP(af0);
00118         archive& BOOST_SERIALIZATION_NVP(af1);
00119         archive& BOOST_SERIALIZATION_NVP(af2);
00120         archive& BOOST_SERIALIZATION_NVP(WN);
00121         archive& BOOST_SERIALIZATION_NVP(tow);
00122         archive& BOOST_SERIALIZATION_NVP(satClkDrift);
00123         archive& BOOST_SERIALIZATION_NVP(dtr);
00124
00125         archive& BOOST_SERIALIZATION_NVP(IOD_ephemeris);
00126         archive& BOOST_SERIALIZATION_NVP(IOD_nav);
00127         archive& BOOST_SERIALIZATION_NVP(SISA);
00128         archive& BOOST_SERIALIZATION_NVP(E5a_HS);
00129         archive& BOOST_SERIALIZATION_NVP(E5b_HS);
00130         archive& BOOST_SERIALIZATION_NVP(E1B_HS);
00131         archive& BOOST_SERIALIZATION_NVP(E5a_DVS);
00132         archive& BOOST_SERIALIZATION_NVP(E5b_DVS);
00133         archive& BOOST_SERIALIZATION_NVP(E1B_DVS);
00134         archive& BOOST_SERIALIZATION_NVP(BGD_E1E5a);
00135         archive& BOOST_SERIALIZATION_NVP(BGD_E1E5b);
00136         archive& BOOST_SERIALIZATION_NVP(flag_all_ephemeris);
00137         if (version > 0)
00138             {
00139                 uint32_t nav_message_type_value = static_cast<uint32_t>(nav_message_type);
00140                 archive& boost::serialization::make_nvp("nav_message_type", nav_message_type_value);
00141                 if (nav_message_type_value <= static_cast<uint32_t>(Galileo_Nav_Message_Type::FNAV))
00142                     {
00143                         nav_message_type =
      static_cast<Galileo_Nav_Message_Type>(nav_message_type_value);
00144                     }
00145                 else
00146                     {
00147                         nav_message_type = Galileo_Nav_Message_Type::Unknown;
00148                     }
00149             }
00150         if (version > 1)
00151             {
00152                 uint32_t nav_message_source_value = static_cast<uint32_t>(nav_message_source);
00153                 archive& boost::serialization::make_nvp("nav_message_source",
      nav_message_source_value);
00154                 if (nav_message_source_value <=
```

```
      static_cast<uint32_t>(Galileo_Nav_Message_Source::E5b))
00155                     {
00156                         nav_message_source =
      static_cast<Galileo_Nav_Message_Source>(nav_message_source_value);
00157                     }
00158                 else
00159                     {
00160                         nav_message_source = Galileo_Nav_Message_Source::Unknown;
00161                     }
00162             }
00163     }
00164 };
00165
00166
00167 BOOST_CLASS_VERSION(Galileo_Ephemeris, 2)
00168
00169
00170 /** \} */
00171 /** \} */
00172 #endif  // GNSS_SDR_GALILEO_EPHEMERIS_H
```

13.704 Galileo_FNAV.h File Reference

Galileo FNAV message constants.

```
#include "MATH_CONSTANTS.h"
#include <cstdint>
#include <utility>
#include <vector>
```

Functions

- const std::vector< std::pair< int32_t, int32_t > > **FNAV_PAGE_TYPE_BIT** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SV_ID_PRN_1_BIT** ({{7, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_1_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0C_1_BIT** ({{23, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_1_BIT** ({{37, 31}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_1_BIT** ({{68, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF2_1_BIT** ({{89, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SISA_1_BIT** ({{95, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI0_1_BIT** ({{103, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI1_1_BIT** ({{114, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AI2_1_BIT** ({{125, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION1_1_BIT** ({{139, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION2_1_BIT** ({{140, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION3_1_BIT** ({{141, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION4_1_BIT** ({{142, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_REGION5_1_BIT** ({{143, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_BGD_1_BIT** ({{144, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_1_BIT** ({{154, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_1_BIT** ({{156, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_1_BIT** ({{168, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5ADVS_1_BIT** ({{188, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_2_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_2_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_2_BIT** ({{49, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_2_BIT** ({{73, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A12_2_BIT** ({{105, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_2_BIT** ({{137, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IDOT_2_BIT** ({{169, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_2_BIT** ({{183, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_2_BIT** ({{195, 20}})

- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_3_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_I0_3_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_3_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAN_3_BIT** ({{81, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CUC_3_BIT** ({{97, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CUS_3_BIT** ({{113, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CRC_3_BIT** ({{129, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CRS_3_BIT** ({{145, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0E_3_BIT** ({{161, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_WN_3_BIT** ({{175, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_3_BIT** ({{187, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DNAV_4_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CIC_4_BIT** ({{17, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_CIS_4_BIT** ({{33, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A0_4_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A1_4_BIT** ({{81, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTATLS_4_BIT** ({{105, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0T_4_BIT** ({{113, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NOT_4_BIT** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NLSF_4_BIT** ({{129, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DN_4_BIT** ({{137, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTATLSF_4_BIT** ({{140, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0G_4_BIT** ({{148, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A0G_4_BIT** ({{156, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_A1G_4_BIT** ({{172, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_N0G_4_BIT** ({{184, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_TOW_4_BIT** ({{190, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DA_5_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_NA_5_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_T0A_5_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D1_5_BIT** ({{23, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_1_5_BIT** ({{29, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_1_5_BIT** ({{42, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_1_5_BIT** ({{53, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_1_5_BIT** ({{69, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_1_5_BIT** ({{80, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_1_5_BIT** ({{96, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_1_5_BIT** ({{107, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_1_5_BIT** ({{123, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_1_5_BIT** ({{139, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_1_5_BIT** ({{152, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D2_5_BIT** ({{154, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_2_5_BIT** ({{160, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_2_5_BIT** ({{173, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_2_5_BIT** ({{184, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_2_5_BIT** ({{200, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_IO_DA_6_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_2_6_BIT** ({{23, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_2_6_BIT** ({{34, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_2_6_BIT** ({{50, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_2_6_BIT** ({{66, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_2_6_BIT** ({{79, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_SVI_D3_6_BIT** ({{81, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAA12_3_6_BIT** ({{87, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E_3_6_BIT** ({{100, 11}})

- const std::vector< std::pair< int32_t, int32_t > > **FNAV_W_3_6_BIT** ({{111, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_DELTAI_3_6_BIT** ({{127, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGA0_3_6_BIT** ({{138, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_OMEGADOT_3_6_BIT** ({{154, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_M0_3_6_BIT** ({{165, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF0_3_6_BIT** ({{181, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_AF1_3_6_BIT** ({{197, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **FNAV_E5AHS_3_6_BIT** ({{210, 2}})

Variables

- constexpr int32_t FNAV_T0C_1_LSB = 60
- constexpr double FNAV_AF0_1_LSB = TWO_N34
- constexpr double FNAV_AF1_1_LSB = TWO_N46
- constexpr double FNAV_AF2_1_LSB = TWO_N59
- constexpr double FNAV_AI0_1_LSB = TWO_N2
- constexpr double FNAV_AI1_1_LSB = TWO_N8
- constexpr double FNAV_AI2_1_LSB = TWO_N15
- constexpr double FNAV_BGD_1_LSB = TWO_N32
- constexpr double FNAV_M0_2_LSB = PI_TWO_N31
- constexpr double FNAV_OMEGADOT_2_LSB = PI_TWO_N43
- constexpr double FNAV_E_2_LSB = TWO_N33
- constexpr double FNAV_A12_2_LSB = TWO_N19
- constexpr double FNAV_OMEGA0_2_LSB = PI_TWO_N31
- constexpr double FNAV_IDOT_2_LSB = PI_TWO_N43
- constexpr double FNAV_I0_3_LSB = PI_TWO_N31
- constexpr double FNAV_W_3_LSB = PI_TWO_N31
- constexpr double FNAV_DELTAN_3_LSB = PI_TWO_N43
- constexpr double FNAV_CUC_3_LSB = TWO_N29
- constexpr double FNAV_CUS_3_LSB = TWO_N29
- constexpr double FNAV_CRC_3_LSB = TWO_N5
- constexpr double FNAV_CRS_3_LSB = TWO_N5
- constexpr int32_t FNAV_T0E_3_LSB = 60
- constexpr double FNAV_CIC_4_LSB = TWO_N29
- constexpr double FNAV_CIS_4_LSB = TWO_N29
- constexpr double FNAV_A0_4_LSB = TWO_N30
- constexpr double FNAV_A1_4_LSB = TWO_N50
- constexpr int32_t FNAV_T0T_4_LSB = 3600
- constexpr int32_t FNAV_T0G_4_LSB = 3600
- constexpr double FNAV_A0G_4_LSB = TWO_N35
- constexpr double FNAV_A1G_4_LSB = TWO_N51
- constexpr int32_t FNAV_T0A_5_LSB = 600
- constexpr double FNAV_DELTAA12_5_LSB = TWO_N9
- constexpr double FNAV_E_5_LSB = TWO_N16
- constexpr double FNAV_W_5_LSB = TWO_N15
- constexpr double FNAV_DELTAI_5_LSB = TWO_N14
- constexpr double FNAV_OMEGA0_5_LSB = TWO_N15
- constexpr double FNAV_OMEGADOT_5_LSB = TWO_N33
- constexpr double FNAV_M0_5_LSB = TWO_N15
- constexpr double FNAV_AF0_5_LSB = TWO_N19
- constexpr double FNAV_AF1_5_LSB = TWO_N38

13.704.1 Detailed Description

Galileo FNAV message constants.

Author

Carles Fernandez, 2020. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_FNAV.h.

13.705 Galileo_FNAV.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_FNAV.h
00003  * \brief Galileo FNAV message constants
00004  * \author Carles Fernandez, 2020. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GALILEO_FNAV_H
00019 #define GNSS_SDR_GALILEO_FNAV_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include <cstdint>
00023 #include <utility>
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 const std::vector<std::pair<int32_t, int32_t» FNAV_PAGE_TYPE_BIT({{1, 6}});
00033
00034 /* WORD 1 iono corrections. FNAV (Galileo E5a message)*/
00035 const std::vector<std::pair<int32_t, int32_t» FNAV_SV_ID_PRN_1_BIT({{7, 6}});
00036 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DNAV_1_BIT({{13, 10}});
00037 const std::vector<std::pair<int32_t, int32_t» FNAV_T0C_1_BIT({{23, 14}});
00038 constexpr int32_t FNAV_T0C_1_LSB = 60;
00039 const std::vector<std::pair<int32_t, int32_t» FNAV_AF0_1_BIT({{37, 31}});
00040 constexpr double FNAV_AF0_1_LSB = TWO_N34;
00041 const std::vector<std::pair<int32_t, int32_t» FNAV_AF1_1_BIT({{68, 21}});
00042 constexpr double FNAV_AF1_1_LSB = TWO_N46;
00043 const std::vector<std::pair<int32_t, int32_t» FNAV_AF2_1_BIT({{89, 6}});
00044 constexpr double FNAV_AF2_1_LSB = TWO_N59;
00045 const std::vector<std::pair<int32_t, int32_t» FNAV_SISA_1_BIT({{95, 8}});
00046 const std::vector<std::pair<int32_t, int32_t» FNAV_AI0_1_BIT({{103, 11}});
00047 constexpr double FNAV_AI0_1_LSB = TWO_N2;
00048 const std::vector<std::pair<int32_t, int32_t» FNAV_AI1_1_BIT({{114, 11}});
00049 constexpr double FNAV_AI1_1_LSB = TWO_N8;
00050 const std::vector<std::pair<int32_t, int32_t» FNAV_AI2_1_BIT({{125, 14}});
00051 constexpr double FNAV_AI2_1_LSB = TWO_N15;
00052 const std::vector<std::pair<int32_t, int32_t» FNAV_REGION1_1_BIT({{139, 1}});
00053 const std::vector<std::pair<int32_t, int32_t» FNAV_REGION2_1_BIT({{140, 1}});
00054 const std::vector<std::pair<int32_t, int32_t» FNAV_REGION3_1_BIT({{141, 1}});
00055 const std::vector<std::pair<int32_t, int32_t» FNAV_REGION4_1_BIT({{142, 1}});
00056 const std::vector<std::pair<int32_t, int32_t» FNAV_REGION5_1_BIT({{143, 1}});
00057 const std::vector<std::pair<int32_t, int32_t» FNAV_BGD_1_BIT({{144, 10}});
00058 constexpr double FNAV_BGD_1_LSB = TWO_N32;
00059 const std::vector<std::pair<int32_t, int32_t» FNAV_E5AHS_1_BIT({{154, 2}});
00060 const std::vector<std::pair<int32_t, int32_t» FNAV_WN_1_BIT({{156, 12}});
00061 const std::vector<std::pair<int32_t, int32_t» FNAV_TOW_1_BIT({{168, 20}});
00062 const std::vector<std::pair<int32_t, int32_t» FNAV_E5ADVS_1_BIT({{188, 1}});
00063
00064 // WORD 2 Ephemeris (1/3)
00065 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DNAV_2_BIT({{7, 10}});
00066 const std::vector<std::pair<int32_t, int32_t» FNAV_M0_2_BIT({{17, 32}});
00067 constexpr double FNAV_M0_2_LSB = PI_TWO_N31;
```

```
00068 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGADOT_2_BIT({{49, 24}});
00069 constexpr double FNAV_OMEGADOT_2_LSB = PI_TWO_N43;
00070 const std::vector<std::pair<int32_t, int32_t» FNAV_E_2_BIT({{73, 32}});
00071 constexpr double FNAV_E_2_LSB = TWO_N33;
00072 const std::vector<std::pair<int32_t, int32_t» FNAV_A12_2_BIT({{105, 32}});
00073 constexpr double FNAV_A12_2_LSB = TWO_N19;
00074 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGA0_2_BIT({{137, 32}});
00075 constexpr double FNAV_OMEGA0_2_LSB = PI_TWO_N31;
00076 const std::vector<std::pair<int32_t, int32_t» FNAV_IDOT_2_BIT({{169, 14}});
00077 constexpr double FNAV_IDOT_2_LSB = PI_TWO_N43;
00078 const std::vector<std::pair<int32_t, int32_t» FNAV_WN_2_BIT({{183, 12}});
00079 const std::vector<std::pair<int32_t, int32_t» FNAV_TOW_2_BIT({{195, 20}});
00080
00081 // WORD 3 Ephemeris (2/3)
00082 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DNAV_3_BIT({{7, 10}});
00083 const std::vector<std::pair<int32_t, int32_t» FNAV_I0_3_BIT({{17, 32}});
00084 constexpr double FNAV_I0_3_LSB = PI_TWO_N31;
00085 const std::vector<std::pair<int32_t, int32_t» FNAV_W_3_BIT({{49, 32}});
00086 constexpr double FNAV_W_3_LSB = PI_TWO_N31;
00087 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAN_3_BIT({{81, 16}});
00088 constexpr double FNAV_DELTAN_3_LSB = PI_TWO_N43;
00089 const std::vector<std::pair<int32_t, int32_t» FNAV_CUC_3_BIT({{97, 16}});
00090 constexpr double FNAV_CUC_3_LSB = TWO_N29;
00091 const std::vector<std::pair<int32_t, int32_t» FNAV_CUS_3_BIT({{113, 16}});
00092 constexpr double FNAV_CUS_3_LSB = TWO_N29;
00093 const std::vector<std::pair<int32_t, int32_t» FNAV_CRC_3_BIT({{129, 16}});
00094 constexpr double FNAV_CRC_3_LSB = TWO_N5;
00095 const std::vector<std::pair<int32_t, int32_t» FNAV_CRS_3_BIT({{145, 16}});
00096 constexpr double FNAV_CRS_3_LSB = TWO_N5;
00097 const std::vector<std::pair<int32_t, int32_t» FNAV_T0E_3_BIT({{161, 14}});
00098 constexpr int32_t FNAV_T0E_3_LSB = 60;
00099 const std::vector<std::pair<int32_t, int32_t» FNAV_WN_3_BIT({{175, 12}});
00100 const std::vector<std::pair<int32_t, int32_t» FNAV_TOW_3_BIT({{187, 20}});
00101
00102 // WORD 4 Ephemeris (3/3)
00103 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DNAV_4_BIT({{7, 10}});
00104 const std::vector<std::pair<int32_t, int32_t» FNAV_CIC_4_BIT({{17, 16}});
00105 constexpr double FNAV_CIC_4_LSB = TWO_N29;
00106 const std::vector<std::pair<int32_t, int32_t» FNAV_CIS_4_BIT({{33, 16}});
00107 constexpr double FNAV_CIS_4_LSB = TWO_N29;
00108 const std::vector<std::pair<int32_t, int32_t» FNAV_A0_4_BIT({{49, 32}});
00109 constexpr double FNAV_A0_4_LSB = TWO_N30;
00110 const std::vector<std::pair<int32_t, int32_t» FNAV_A1_4_BIT({{81, 24}});
00111 constexpr double FNAV_A1_4_LSB = TWO_N50;
00112 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTATLS_4_BIT({{105, 8}});
00113 const std::vector<std::pair<int32_t, int32_t» FNAV_T0T_4_BIT({{113, 8}});
00114 constexpr int32_t FNAV_T0T_4_LSB = 3600;
00115 const std::vector<std::pair<int32_t, int32_t» FNAV_W_NOT_4_BIT({{121, 8}});
00116 const std::vector<std::pair<int32_t, int32_t» FNAV_W_NLSF_4_BIT({{129, 8}});
00117 const std::vector<std::pair<int32_t, int32_t» FNAV_DN_4_BIT({{137, 3}});
00118 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTATLSF_4_BIT({{140, 8}});
00119 const std::vector<std::pair<int32_t, int32_t» FNAV_T0G_4_BIT({{148, 8}});
00120 constexpr int32_t FNAV_T0G_4_LSB = 3600;
00121 const std::vector<std::pair<int32_t, int32_t» FNAV_A0G_4_BIT({{156, 16}});
00122 constexpr double FNAV_A0G_4_LSB = TWO_N35;
00123 const std::vector<std::pair<int32_t, int32_t» FNAV_A1G_4_BIT({{172, 12}});
00124 constexpr double FNAV_A1G_4_LSB = TWO_N51;
00125 const std::vector<std::pair<int32_t, int32_t» FNAV_W_N0G_4_BIT({{184, 6}});
00126 const std::vector<std::pair<int32_t, int32_t» FNAV_TOW_4_BIT({{190, 20}});
00127
00128 // WORD 5 Almanac SVID1 SVID2(1/2)
00129 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DA_5_BIT({{7, 4}});
00130 const std::vector<std::pair<int32_t, int32_t» FNAV_W_NA_5_BIT({{11, 2}});
00131 const std::vector<std::pair<int32_t, int32_t» FNAV_T0A_5_BIT({{13, 10}});
00132 constexpr int32_t FNAV_T0A_5_LSB = 600;
00133 const std::vector<std::pair<int32_t, int32_t» FNAV_SVI_D1_5_BIT({{23, 6}});
00134 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAA12_1_5_BIT({{29, 13}});
00135 constexpr double FNAV_DELTAA12_5_LSB = TWO_N9;
00136 const std::vector<std::pair<int32_t, int32_t» FNAV_E_1_5_BIT({{42, 11}});
00137 constexpr double FNAV_E_5_LSB = TWO_N16;
00138 const std::vector<std::pair<int32_t, int32_t» FNAV_W_1_5_BIT({{53, 16}});
00139 constexpr double FNAV_W_5_LSB = TWO_N15;
00140 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAI_1_5_BIT({{69, 11}});
00141 constexpr double FNAV_DELTAI_5_LSB = TWO_N14;
00142 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGA0_1_5_BIT({{80, 16}});
00143 constexpr double FNAV_OMEGA0_5_LSB = TWO_N15;
00144 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGADOT_1_5_BIT({{96, 11}});
00145 constexpr double FNAV_OMEGADOT_5_LSB = TWO_N33;
00146 const std::vector<std::pair<int32_t, int32_t» FNAV_M0_1_5_BIT({{107, 16}});
00147 constexpr double FNAV_M0_5_LSB = TWO_N15;
00148 const std::vector<std::pair<int32_t, int32_t» FNAV_AF0_1_5_BIT({{123, 16}});
00149 constexpr double FNAV_AF0_5_LSB = TWO_N19;
00150 const std::vector<std::pair<int32_t, int32_t» FNAV_AF1_1_5_BIT({{139, 13}});
00151 constexpr double FNAV_AF1_5_LSB = TWO_N38;
00152 const std::vector<std::pair<int32_t, int32_t» FNAV_E5AHS_1_5_BIT({{152, 2}});
00153 const std::vector<std::pair<int32_t, int32_t» FNAV_SVI_D2_5_BIT({{154, 6}});
00154 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAA12_2_5_BIT({{160, 13}});
```

```
00155 const std::vector<std::pair<int32_t, int32_t» FNAV_E_2_5_BIT({{173, 11}});
00156 const std::vector<std::pair<int32_t, int32_t» FNAV_W_2_5_BIT({{184, 16}});
00157 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAI_2_5_BIT({{200, 11}});
00158 // const std::vector<std::pair<int,int» FNAV_Omega012_2_5_bit({{210,4}});
00159
00160 // WORD 6 Almanac SVID2(1/2) SVID3
00161 const std::vector<std::pair<int32_t, int32_t» FNAV_IO_DA_6_BIT({{7, 4}});
00162 // const std::vector<std::pair<int,int» FNAV_Omega022_2_6_bit({{10,12}});
00163 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGADOT_2_6_BIT({{23, 11}});
00164 const std::vector<std::pair<int32_t, int32_t» FNAV_M0_2_6_BIT({{34, 16}});
00165 const std::vector<std::pair<int32_t, int32_t» FNAV_AF0_2_6_BIT({{50, 16}});
00166 const std::vector<std::pair<int32_t, int32_t» FNAV_AF1_2_6_BIT({{66, 13}});
00167 const std::vector<std::pair<int32_t, int32_t» FNAV_E5AHS_2_6_BIT({{79, 2}});
00168 const std::vector<std::pair<int32_t, int32_t» FNAV_SVI_D3_6_BIT({{81, 6}});
00169 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAA12_3_6_BIT({{87, 13}});
00170 const std::vector<std::pair<int32_t, int32_t» FNAV_E_3_6_BIT({{100, 11}});
00171 const std::vector<std::pair<int32_t, int32_t» FNAV_W_3_6_BIT({{111, 16}});
00172 const std::vector<std::pair<int32_t, int32_t» FNAV_DELTAI_3_6_BIT({{127, 11}});
00173 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGA0_3_6_BIT({{138, 16}});
00174 const std::vector<std::pair<int32_t, int32_t» FNAV_OMEGADOT_3_6_BIT({{154, 11}});
00175 const std::vector<std::pair<int32_t, int32_t» FNAV_M0_3_6_BIT({{165, 16}});
00176 const std::vector<std::pair<int32_t, int32_t» FNAV_AF0_3_6_BIT({{181, 16}});
00177 const std::vector<std::pair<int32_t, int32_t» FNAV_AF1_3_6_BIT({{197, 13}});
00178 const std::vector<std::pair<int32_t, int32_t» FNAV_E5AHS_3_6_BIT({{210, 2}});
00179
00180
00181 /** \} */
00182 /** \} */
00183 #endif  // GNSS_SDR_GALILEO_FNAV_H
```

13.706 galileo_fnav_message.h File Reference

Implementation of a Galileo F/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015).

```
#include "Galileo_E5a.h"
#include "Galileo_FNAV.h"
#include "galileo_almanac_helper.h"
#include "galileo_ephemeris.h"
#include "galileo_iono.h"
#include "galileo_utc_model.h"
#include <bitset>
#include <cstdint>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Galileo_Fnav_Message

  This class handles the Galileo F/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/↩ sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.

13.706.1 Detailed Description

Implementation of a Galileo F/NAV Data message as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015).

Author

Marc Sales, 2014. marcsales92(at)gmail.com \based on work from:

- Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_fnav_message.h.

13.707 galileo_fnav_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_fnav_message.h
00003  * \brief  Implementation of a Galileo F/NAV Data message
00004  *         as described in Galileo OS SIS ICD Issue 1.2 (Nov. 2015)
00005  * \author Marc Sales, 2014. marcsales92(at)gmail.com
00006  * \based on work from:
00007  * <ul>
00008  * <li> Javier Arribas, 2011. jarribas(at)cttc.es
00009  * </ul>
00010  *
00011  *
00012  * -----------------------------------------------------------------------------
00013  *
00014  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00015  * This file is part of GNSS-SDR.
00016  *
00017  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00018  * SPDX-License-Identifier: GPL-3.0-or-later
00019  *
00020  * -----------------------------------------------------------------------------
00021  */
00022
00023 #ifndef GNSS_SDR_GALILEO_FNAV_MESSAGE_H
00024 #define GNSS_SDR_GALILEO_FNAV_MESSAGE_H
00025
00026
00027 #include "Galileo_E5a.h"
00028 #include "Galileo_FNAV.h"
00029 #include "galileo_almanac_helper.h"
00030 #include "galileo_ephemeris.h"
00031 #include "galileo_iono.h"
00032 #include "galileo_utc_model.h"
00033 #include <bitset>
00034 #include <cstdint>
00035 #include <string>
00036 #include <utility>
00037 #include <vector>
00038
00039 /** \addtogroup Core
00040  * \{ */
00041 /** \addtogroup System_Parameters
00042  * \{ */
00043
00044
00045 /*!
00046  * \brief This class handles the Galileo F/NAV Data message, as described in the
00047  * Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan.
      2021).
00048  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
00049  */
00050 class Galileo_Fnav_Message
00051 {
00052 public:
00053     Galileo_Fnav_Message() = default;
00054
00055     void split_page(const std::string& page_string);
00056     bool have_new_ephemeris();
00057     bool have_new_iono_and_GST();
00058     bool have_new_utc_model();
00059     bool have_new_almanac();
00060     Galileo_Ephemeris get_ephemeris() const;
00061     Galileo_Iono get_iono() const;
00062     Galileo_Utc_Model get_utc_model() const;
00063     Galileo_Almanac_Helper get_almanac() const;
00064
00065     inline int32_t get_TOW1() const
00066     {
00067         return FNAV_TOW_1;
00068     }
00069
00070     inline int32_t get_WN1() const
00071     {
00072         return FNAV_WN_1;
00073     }
00074
00075     inline int32_t get_TOW2() const
00076     {
00077         return FNAV_TOW_2;
00078     }
00079
00080     inline int32_t get_WN2() const
00081     {
00082         return FNAV_WN_2;
```

```
00083     }
00084
00085     inline int32_t get_TOW3() const
00086     {
00087         return FNAV_TOW_3;
00088     }
00089
00090     inline int32_t get_WN3() const
00091     {
00092         return FNAV_WN_3;
00093     }
00094
00095     inline int32_t get_TOW4() const
00096     {
00097         return FNAV_TOW_4;
00098     }
00099
00100     inline bool get_flag_CRC_test() const
00101     {
00102         return flag_CRC_test;
00103     }
00104
00105     inline bool get_flag_TOW_set() const
00106     {
00107         return flag_TOW_set;
00108     }
00109
00110     inline void set_flag_TOW_set(bool flag_tow)
00111     {
00112         flag_TOW_set = flag_tow;
00113     }
00114
00115     inline bool is_TOW1_set() const
00116     {
00117         return flag_TOW_1;
00118     }
00119
00120     inline void set_TOW1_flag(bool flag_tow1)
00121     {
00122         flag_TOW_1 = flag_tow1;
00123     }
00124
00125     inline bool is_TOW2_set() const
00126     {
00127         return flag_TOW_2;
00128     }
00129
00130     inline void set_TOW2_flag(bool flag_tow2)
00131     {
00132         flag_TOW_2 = flag_tow2;
00133     }
00134
00135     inline bool is_TOW3_set() const
00136     {
00137         return flag_TOW_3;
00138     }
00139
00140     inline void set_TOW3_flag(bool flag_tow3)
00141     {
00142         flag_TOW_3 = flag_tow3;
00143     }
00144
00145     inline bool is_TOW4_set() const
00146     {
00147         return flag_TOW_4;
00148     }
00149
00150     inline void set_TOW4_flag(bool flag_tow4)
00151     {
00152         flag_TOW_4 = flag_tow4;
00153     }
00154
00155 private:
00156     bool CRC_test(const std::bitset<GALILEO_FNAV_DATA_FRAME_BITS>& bits, uint32_t checksum) const;
00157     void decode_page(const std::string& data);
00158     uint64_t read_navigation_unsigned(const std::bitset<GALILEO_FNAV_DATA_FRAME_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00159     int64_t read_navigation_signed(const std::bitset<GALILEO_FNAV_DATA_FRAME_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00160
00161     std::string omega0_1{};
00162     // std::string omega0_2{};
00163     // bool omega_flag{};
00164
00165     int32_t IOD_ephemeris{};
00166
00167     int32_t page_type{};
```

```
00168     // WORD 1 SVID, Clock correction, SISA, Ionospheric correction, BGD, GST, Signal
00169     // health and Data validity status
00170     int32_t FNAV_SV_ID_PRN_1{};
00171     int32_t FNAV_IODnav_1{};
00172     int32_t FNAV_t0c_1{};
00173     double FNAV_af0_1{};
00174     double FNAV_af1_1{};
00175     double FNAV_af2_1{};
00176     int32_t FNAV_SISA_1{};
00177     double FNAV_ai0_1{};
00178     double FNAV_ai1_1{};
00179     double FNAV_ai2_1{};
00180     double FNAV_BGD_1{};
00181     int32_t FNAV_E5ahs_1{};
00182     int32_t FNAV_WN_1{};
00183     int32_t FNAV_TOW_1{};
00184     bool FNAV_region1_1{};
00185     bool FNAV_region2_1{};
00186     bool FNAV_region3_1{};
00187     bool FNAV_region4_1{};
00188     bool FNAV_region5_1{};
00189     bool FNAV_E5advs_1{};
00190
00191     // WORD 2 Ephemeris (1/3) and GST
00192     int32_t FNAV_IODnav_2{};
00193     double FNAV_M0_2{};
00194     double FNAV_omegadot_2{};
00195     double FNAV_e_2{};
00196     double FNAV_a12_2{};
00197     double FNAV_omega0_2{};
00198     double FNAV_idot_2{};
00199     int32_t FNAV_WN_2{};
00200     int32_t FNAV_TOW_2{};
00201
00202     // WORD 3 Ephemeris (2/3) and GST
00203     int32_t FNAV_IODnav_3{};
00204     double FNAV_i0_3{};
00205     double FNAV_w_3{};
00206     double FNAV_deltan_3{};
00207     double FNAV_Cuc_3{};
00208     double FNAV_Cus_3{};
00209     double FNAV_Crc_3{};
00210     double FNAV_Crs_3{};
00211     int32_t FNAV_t0e_3{};
00212     int32_t FNAV_WN_3{};
00213     int32_t FNAV_TOW_3{};
00214
00215     // WORD 4 Ephemeris (3/3), GST-UTC conversion, GST-GPS conversion and TOW.
00216     // Note that the clock is repeated in this page type
00217     int32_t FNAV_IODnav_4{};
00218     double FNAV_Cic_4{};
00219     double FNAV_Cis_4{};
00220     double FNAV_A0_4{};
00221     double FNAV_A1_4{};
00222     int32_t FNAV_deltatls_4{};
00223     int32_t FNAV_t0t_4{};
00224     int32_t FNAV_WNot_4{};
00225     int32_t FNAV_WNlsf_4{};
00226     int32_t FNAV_DN_4{};
00227     int32_t FNAV_deltatlsf_4{};
00228     int32_t FNAV_t0g_4{};
00229     double FNAV_A0g_4{};
00230     double FNAV_A1g_4{};
00231     int32_t FNAV_WN0g_4{};
00232     int32_t FNAV_TOW_4{};
00233
00234     // WORD 5 Almanac (SVID1 and SVID2(1/2)), Week Number and almanac reference time
00235     int32_t FNAV_IODa_5{};
00236     int32_t FNAV_WNa_5{};
00237     int32_t FNAV_t0a_5{};
00238     int32_t FNAV_SVID1_5{};
00239     double FNAV_Deltaa12_1_5{};
00240     double FNAV_e_1_5{};
00241     double FNAV_w_1_5{};
00242     double FNAV_deltai_1_5{};
00243     double FNAV_Omega0_1_5{};
00244     double FNAV_Omegadot_1_5{};
00245     double FNAV_M0_1_5{};
00246     double FNAV_af0_1_5{};
00247     double FNAV_af1_1_5{};
00248     uint32_t FNAV_E5ahs_1_5{};
00249     int32_t FNAV_SVID2_5{};
00250     double FNAV_Deltaa12_2_5{};
00251     double FNAV_e_2_5{};
00252     double FNAV_w_2_5{};
00253     double FNAV_deltai_2_5{};
00254
```

```
00255     // WORD 6 Almanac (SVID2(2/2) and SVID3)
00256     int32_t FNAV_IODa_6{};
00257     double FNAV_Omega0_2_6{};
00258     double FNAV_Omegadot_2_6{};
00259     double FNAV_M0_2_6{};
00260     double FNAV_af0_2_6{};
00261     double FNAV_af1_2_6{};
00262     int32_t FNAV_E5ahs_2_6{};
00263     int32_t FNAV_SVID3_6{};
00264     double FNAV_Deltaa12_3_6{};
00265     double FNAV_e_3_6{};
00266     double FNAV_w_3_6{};
00267     double FNAV_deltai_3_6{};
00268     double FNAV_Omega0_3_6{};
00269     double FNAV_Omegadot_3_6{};
00270     double FNAV_M0_3_6{};
00271     double FNAV_af0_3_6{};
00272     double FNAV_af1_3_6{};
00273     int32_t FNAV_E5ahs_3_6{};
00274
00275     bool flag_CRC_test{};
00276     bool flag_all_ephemeris{};  // Flag indicating that all words containing ephemeris have been
      received
00277     bool flag_ephemeris_1{};    // Flag indicating that ephemeris 1/3 (word 2) have been received
00278     bool flag_ephemeris_2{};    // Flag indicating that ephemeris 2/3 (word 3) have been received
00279     bool flag_ephemeris_3{};    // Flag indicating that ephemeris 3/3 (word 4) have been received
00280
00281     bool flag_iono_and_GST{};  // Flag indicating that ionospheric and GST parameters (word 1) have
      been received
00282     bool flag_iono_model_valid{};
00283     bool flag_TOW_1{};
00284     bool flag_TOW_2{};
00285     bool flag_TOW_3{};
00286     bool flag_TOW_4{};
00287     bool flag_TOW_set{};          // it is true when page 1,2,3 or 4 arrives
00288     bool flag_utc_model_valid{};  // Flag indicating that UTC model parameters (word 4) have been
      received
00289     bool flag_new_utc_model{};    // Flag indicating that a newly decoded UTC model is pending
      publication
00290
00291     bool flag_all_almanac{};  // Flag indicating that all Almanac data have been received
00292     bool flag_almanac_1{};    // Flag indicating that almanac 1/2 (word 5) have been received
00293     bool flag_almanac_2{};    // Flag indicating that almanac 2/2 (word 6) have been received
00294 };
00295
00296
00297 /** \} */
00298 /** \} */
00299 #endif  // GNSS_SDR_GALILEO_FNAV_MESSAGE_H
```

13.708 galileo_has_data.h File Reference

Class for Galileo HAS message type 1 data storage.

```
#include "Galileo_CNAV.h"
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- struct mt1_header
- class Galileo_HAS_data

  This class is a storage for Galileo HAS message type 1, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022. See https://www.gsc-europa.eu/↩ sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf.

13.708.1 Detailed Description

Class for Galileo HAS message type 1 data storage.

Author

Carles Fernandez-Prades, 2020-2022 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_has_data.h.

13.709 galileo_has_data.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_has_data.h
00003  * \brief Class for Galileo HAS message type 1 data storage
00004  * \author Carles Fernandez-Prades, 2020-2022 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GALILEO_HAS_DATA_H
00019 #define GNSS_SDR_GALILEO_HAS_DATA_H
00020
00021 #include "Galileo_CNAV.h"  // for GALILEO_HAS_INVALID_TOW, GALILEO_HAS_INVALID_WEEK
00022 #include <cstdint>
00023 #include <string>
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031 struct mt1_header
00032 {
00033     uint16_t toh;
00034     uint8_t mask_id;
00035     uint8_t iod_set_id;
00036     uint8_t reserved;
00037     bool mask_flag;
00038     bool orbit_correction_flag;
00039     bool clock_fullset_flag;
00040     bool clock_subset_flag;
00041     bool code_bias_flag;
00042     bool phase_bias_flag;
00043 };
00044
00045 /*!
00046  * \brief This class is a storage for Galileo HAS message type 1, as defined in
00047  * Galileo High Accuracy Service Signal-In-Space Interface Control Document
00048  * (HAS SIS ICD) Issue 1.0, May 2022.
00049  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf
00050  */
00051 class Galileo_HAS_data
00052 {
00053 public:
00054     Galileo_HAS_data() = default;
00055
00056     std::vector<std::string> get_signals_in_mask(uint8_t nsys) const;                    //!< Get a
      vector of Nsys std::string with signals in mask for system nsys, with 0 <= nsys < Nsys
00057     std::vector<std::string> get_signals_in_mask(const std::string& system) const;       //!< Get a
      vector of Nsys std::string with signals in mask for system ("GPS"/"Galileo")
00058     std::vector<std::string> get_systems_string() const;                                 //!< Get
      Nsys system name strings
00059     std::vector<std::string> get_systems_subset_string() const;                          //!< Get
      Nsat system name strings present in clock corrections subset
00060     std::vector<std::vector<float» get_code_bias_m() const;                             //!< Get Nsat
      x Ncodes code biases in [m]
00061     std::vector<std::vector<float» get_phase_bias_cycle() const;                        //!< Get Nsat
      x Nphases phase biases in [cycles]
00062     std::vector<std::vector<float» get_delta_clock_subset_correction_m() const;         //!< Get
      Nsys_sub vectors with Nsat_sub delta clock C0 corrections in [m]
00063     std::vector<float> get_delta_radial_m() const;                                       //!< Get
      Nsat delta radial corrections in [m]
```

```
00064     std::vector<float> get_delta_radial_m(uint8_t nsys) const;                                  //!< Get
      delta radial corrections in [m] for system nsys, with 0 <= nsys < Nsys
00065     std::vector<float> get_delta_in_track_m() const;                                           //!< Get
      Nsat delta in-track corrections in [m]
00066     std::vector<float> get_delta_in_track_m(uint8_t nsys) const;                               //!< Get
      delta in-track corrections in [m] for system nsys, with 0 <= nsys < Nsys
00067     std::vector<float> get_delta_cross_track_m() const;                                        //!< Get
      Nsat delta cross-track corrections in [m]
00068     std::vector<float> get_delta_cross_track_m(uint8_t nsys) const;                            //!< Get
      delta cross-track corrections in [m] for system nsys, with 0 <= nsys < Nsys
00069     std::vector<float> get_delta_clock_correction_m() const;                                   //!< Get
      Nsat delta clock C0 corrections in [m]
00070     std::vector<float> get_delta_clock_correction_m(uint8_t nsys) const;                       //!< Get
      delta clock C0 corrections in [m] for system nsys, with 0 <= nsys < Nsys
00071     std::vector<float> get_delta_clock_subset_correction_m(uint8_t nsys) const;                //!< Get
      delta clock C0 subset corrections in [m] for system nsys, with 0 <= nsys < Nsys
00072     std::vector<int> get_PRNs_in_mask(uint8_t nsys) const;                                     //!< Get
      PRNs in mask for system nsys, with 0 <= nsys < Nsys
00073     std::vector<int> get_PRNs_in_mask(const std::string& system) const;                        //!< Get
      PRNs in mask for system ("GPS"/"Galileo")
00074     std::vector<int> get_PRNs_in_submask(uint8_t nsys) const;                                  //!< Get
      PRNs in submask for system nsys, with 0 <= nsys < Nsys
00075     std::vector<uint16_t> get_gnss_iod(uint8_t nsys) const;                                    //!< Get
      GNSS IODs for for system nsys, with 0 <= nsys < Nsys
00076     std::vector<uint8_t> get_num_satellites() const;                                           //!< Get
      Nsys number of satellites
00077     std::vector<uint8_t> get_num_subset_satellites() const;                                    //!< Get
      Nsys_sub number of satellites
00078     float get_code_bias_m(const std::string& signal, int PRN) const;                           //!< Get
      code bias in [m] for a given signal and PRN satellite
00079     float get_phase_bias_cycle(const std::string& signal, int PRN) const;                      //!< Get
      phase bias in [cycles] for a given signal and PRN satellite
00080     float get_delta_radial_m(const std::string& system, int prn) const;                        //!< Get
      orbital radial correction in [m] for a given system ("GPS"/"Galileo") and PRN
00081     float get_delta_in_track_m(const std::string& system, int prn) const;                      //!< Get
      orbital in_track correction in [m] for a given system ("GPS"/"Galileo") and PRN
00082     float get_delta_cross_track_m(const std::string& system, int prn) const;                   //!< Get
      orbital cross_track correction in [m] for a given system ("GPS"/"Galileo") and PRN
00083     float get_clock_correction_mult_m(const std::string& system, int prn) const;               //!< Get
      clock correction in [m], already multiplied by its Delta Clock Multiplier, for a given system
      ("GPS"/"Galileo") and PRN
00084     float get_clock_subset_correction_mult_m(const std::string& system, int prn) const;        //!< Get
      clock correction subset in [m], already multiplied by its Delta Clock Multiplier
00085     uint16_t get_nsat() const;                                                                 //!< Get
      total number of satellites with corrections
00086     uint16_t get_nsat_sub() const;                                                             //!< Get
      number of satellites in clock subset corrections
00087     uint16_t get_validity_interval_s(uint8_t validity_interval_index) const;                   //!< Get
      validity interval in [s] from the validity_interval_index
00088     uint32_t get_time_of_message_s() const;                                                    //!< Get
      Time of Message, in [s] of week. See HAS SIS ICD 1.0 Section 7.7
00089     uint16_t get_gnss_iod(const std::string& system, int prn) const;                           //!< Get
      GNSS IOD from a given system ("GPS"/"Galileo") and PRN
00090     uint8_t get_gnss_id(int nsat) const;                                                       //!< Get
      GNSS ID from the nsat satellite
00091     uint8_t get_phase_discontinuity_indicator(const std::string& signal, int PRN) const;       //!< Get PDI
      for a given signal and PRN satellite
00092     static bool has_validity_interval_at_tow(uint32_t valid_until, uint32_t tow);              //!< Check
      whether a validity end time is active at the given TOW
00093     static bool is_orbit_radial_unavailable(float value);                                      //!< Check
      HAS radial correction unavailable sentinel
00094     static bool is_orbit_along_cross_unavailable(float value);                                 //!< Check
      HAS along/cross-track correction unavailable sentinel
00095     static bool is_clock_unavailable(float value);                                             //!< Check
      HAS clock correction unavailable sentinel
00096     static bool is_clock_do_not_use(float value);                                              //!< Check
      HAS clock correction do-not-use sentinel
00097     static bool is_code_bias_unavailable(float value);                                         //!< Check
      HAS code bias unavailable sentinel
00098     static bool is_phase_bias_unavailable(float value);                                        //!< Check
      HAS phase bias unavailable sentinel
00099
00100     // Mask
00101     std::vector<uint8_t> gnss_id_mask;                      //!< GNSS ID. See HAS SIS ICD 1.0 Section
      5.2.1.1
00102     std::vector<uint64_t> satellite_mask;                   //!< SatM - Satellite Mask. See HAS SIS
      ICD 1.0 Section 5.2.1.2
00103     std::vector<uint16_t> signal_mask;                      //!< SigM - Signal Mask. See HAS SIS ICD
      1.0 Section 5.2.1.3
00104     std::vector<bool> cell_mask_availability_flag;          //!< CMAF - Cell Mask Availability Flag.
      See HAS SIS ICD 1.0 Section 5.2.1.4
00105     std::vector<std::vector<std::vector<bool>>> cell_mask;  //!< CM - Cell Mask. See HAS SIS ICD 1.0
      Section 5.2.1.5
00106     std::vector<uint8_t> nav_message;                       //!< NM - Navigation Message Index. See
      HAS SIS ICD 1.0 Section 5.2.1.6
00107
00108     // Orbit corrections
```

```
00109     std::vector<uint16_t> gnss_iod;          //!< IODref - Reference Issue of Data. See HAS SIS ICD
      1.0 Table 26
00110     std::vector<int16_t> delta_radial;       //!< DR - Delta Radial Correction. See HAS SIS ICD 1.0
      Table 25
00111     std::vector<int16_t> delta_in_track;     //!< DIT - Delta In-Track Correction. See HAS SIS ICD 1.0
      Table 25
00112     std::vector<int16_t> delta_cross_track;  //!< DCT - Delta Cross Correction. See HAS SIS ICD 1.0
      Table 25
00113
00114     // Clock full-set corrections
00115     std::vector<uint8_t> delta_clock_multiplier;  //!< DCM - Delta Clock Multipliers. See HAS SIS ICD
      1.0 Section 5.2.3.1
00116     std::vector<int16_t> delta_clock_correction;  //!< DCC - Delta Clock Corrections. See HAS SIS ICD
      1.0 Section 5.2.3.2
00117
00118     // Clock subset corrections
00119     std::vector<uint8_t> gnss_id_clock_subset;                              //!< GNSS ID. Specific
      GNSS to which the corrections refer. See HAS SIS ICD 1.0 Section 5.2.1.1
00120     std::vector<uint8_t> delta_clock_multiplier_clock_subset;               //!< DCM. Multiplier for
      all Delta Clock corrections. See HAS SIS ICD 1.0 Section 5.2.3.1
00121     std::vector<uint64_t> satellite_submask;                                //!< SatMsub - Satellite
      Subset Mask. See HAS SIS ICD 1.0 Section 5.2.4.1
00122     std::vector<std::vector<int16_t> delta_clock_correction_clock_subset;  //!< DCCsub - Delta Clock
      Subset Corrections. See HAS SIS ICD 1.0 Section 5.2.4.1
00123
00124     // Code bias
00125     std::vector<std::vector<int16_t> code_bias;  //!< CB - Code bias for the m-th signal of the n-th
      SV. See HAS SIS ICD 1.0 Section 5.2.5
00126
00127     // Phase bias
00128     std::vector<std::vector<int16_t> phase_bias;                    //!< PB - Phase bias for the m-th
      signal of the n-th SV. See HAS SIS ICD 1.0 Section 5.2.6
00129     std::vector<std::vector<uint8_t> phase_discontinuity_indicator;  //!< PDI - Phase Discontinuity
      Indicator. See HAS SIS ICD 1.0 Section 5.2.6.
00130
00131     uint32_t week{GALILEO_HAS_INVALID_WEEK};  //!< Galileo week number
00132     uint32_t tow{GALILEO_HAS_INVALID_TOW};    //!< Time of Week
00133
00134     mt1_header header;   //!< MT1 Header parameters. See HAS SIS ICD 1.0 Section 5.1.1
00135     uint8_t has_status;  //!< HASS - HAS Status (from HAS page header). See HAS SIS ICD 1.0 Section
      3.1.1
00136     uint8_t message_id;  //!< MID - Message ID (from HAS page header). See HAS SIS ICD 1.0 Section 3.1
00137
00138     uint8_t Nsys;      //!< Number of GNSS for which corrections are provided. See HAS SIS ICD 1.0
      Section 5.2.1
00139     uint8_t Nsys_sub;  //!< Number of GNSS for which corrections are provided in clock subset
      corrections. See HAS SIS ICD 1.0 Section 5.2.2.1
00140
00141     uint8_t validity_interval_index_orbit_corrections;          //!< VI - Validity Interval Index for
      Orbit corrections. See HAS SIS ICD 1.0 Section 5.2.2.1
00142     uint8_t validity_interval_index_clock_fullset_corrections;  //!< VI - Validity Interval Index for
      Clock full-set corrections. See HAS SIS ICD 1.0 Section 5.2.2.1
00143     uint8_t validity_interval_index_clock_subset_corrections;   //!< VI - Validity Interval Index for
      Clock subset corrections. See HAS SIS ICD 1.0 Section 5.2.2.1
00144     uint8_t validity_interval_index_code_bias_corrections;      //!< VI - Validity Interval Index for
      Code bias. See HAS SIS ICD 1.0 Section 5.2.2.1
00145     uint8_t validity_interval_index_phase_bias_corrections;     //!< VI - Validity Interval Index for
      Phase bias. See HAS SIS ICD 1.0 Section 5.2.2.1
00146 };
00147
00148
00149 /** \} */
00150 /** \} */
00151 #endif  // GNSS_SDR_GALILEO_HAS_DATA_H
```

13.710 galileo_has_page.h File Reference

Class for Galileo HAS message page storage.

```
#include "Galileo_CNAV.h"
#include <cstdint>
#include <string>
```

Classes

- class Galileo_HAS_page

  This class is a storage for Galileo HAS message page, as defined in Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD) Issue 1.0, May 2022.

13.710.1 Detailed Description

Class for Galileo HAS message page storage.

Author

Carles Fernandez-Prades, 2021 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2022 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_has_page.h.

13.711 galileo_has_page.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_has_page.h
00003  * \brief Class for Galileo HAS message page storage
00004  * \author Carles Fernandez-Prades, 2021 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2022  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GALILEO_HAS_PAGE_H
00019 #define GNSS_SDR_GALILEO_HAS_PAGE_H
00020
00021 #include "Galileo_CNAV.h"  // for GALILEO_HAS_INVALID_TOW, GALILEO_HAS_INVALID_WEEK
00022 #include <cstdint>
00023 #include <string>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief This class is a storage for Galileo HAS message page, as defined in
00033  * Galileo High Accuracy Service Signal-In-Space Interface Control Document
00034  * (HAS SIS ICD) Issue 1.0, May 2022
00035  */
00036 class Galileo_HAS_page
00037 {
00038 public:
00039     Galileo_HAS_page() = default;
00040
00041     std::string has_message_string;           //!< HAS message content
00042     uint64_t time_stamp{};                    //!< HAS page time stamp, in [s]
00043     uint32_t week{GALILEO_HAS_INVALID_WEEK};  //!< HAS page Galileo week number
00044     uint32_t tow{GALILEO_HAS_INVALID_TOW};    //!< HAS page time of week, in [s]
00045
00046     // HAS page header
00047     uint8_t has_status{};       //!< HAS status
00048     uint8_t reserved{};         //!< HAS reserved field
00049     uint8_t message_type{};     //!< HAS message type (MT)
00050     uint8_t message_id{};       //!< HAS message ID (MID)
00051     uint8_t message_size{};     //!< HAS message size (MS)
00052     uint8_t message_page_id{};  //!< HAS message page ID (PID)
00053 };
00054
00055
00056 /** \} */
00057 /** \} */
00058 #endif  // GNSS_SDR_GALILEO_HAS_PAGE_H
```

13.712 Galileo_INAV.h File Reference

Galileo INAV message constants.

```
#include "MATH_CONSTANTS.h"
#include <cstddef>
#include <cstdint>
#include <utility>
#include <vector>
```

Functions

- const std::vector< std::pair< int32_t, int32_t > > **TYPE** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **PAGE_TYPE_BIT** ({{1, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_1_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **T0_E_1_BIT** ({{17, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_1_BIT** ({{31, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **E_1_BIT** ({{63, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1_BIT** ({{95, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_2_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_0_2_BIT** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **I_0_2_BIT** ({{49, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_2_BIT** ({{81, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **I_DOT_2_BIT** ({{113, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_3_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_3_BIT** ({{17, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_N_3_BIT** ({{41, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_UC_3_BIT** ({{57, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_US_3_BIT** ({{73, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RC_3_BIT** ({{89, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RS_3_BIT** ({{105, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **SISA_3_BIT** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_NAV_4_BIT** ({{7, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_ID_PRN_4_BIT** ({{17, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IC_4_BIT** ({{23, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IS_4_BIT** ({{39, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **T0C_4_BIT** ({{55, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_4_BIT** ({{69, 31}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_4_BIT** ({{100, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **AF2_4_BIT** ({{121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **SPARE_4_BIT** ({{127, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **AI0_5_BIT** ({{7, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **AI1_5_BIT** ({{18, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **AI2_5_BIT** ({{29, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION1_5_BIT** ({{43, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION2_5_BIT** ({{44, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION3_5_BIT** ({{45, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION4_5_BIT** ({{46, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **REGION5_5_BIT** ({{47, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **BGD_E1_E5A_5_BIT** ({{48, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **BGD_E1_E5B_5_BIT** ({{58, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_5_BIT** ({{68, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_5_BIT** ({{70, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_DVS_5_BIT** ({{72, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_DVS_5_BIT** ({{73, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_5_BIT** ({{74, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_5_BIT** ({{86, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **SPARE_5_BIT** ({{106, 23}})

- const std::vector< std::pair< int32_t, int32_t > > **A0_6_BIT** ({{7, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **A1_6_BIT** ({{39, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_LS_6_BIT** ({{63, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T0T_6_BIT** ({{71, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **W_NOT_6_BIT** ({{79, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_LSF_6_BIT** ({{87, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DN_6_BIT** ({{95, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_LSF_6_BIT** ({{98, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_6_BIT** ({{106, 20}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_7_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A_7_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T0A_7_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D1_7_BIT** ({{23, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_7_BIT** ({{29, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_7_BIT** ({{42, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_7_BIT** ({{53, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_7_BIT** ({{69, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_7_BIT** ({{80, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_7_BIT** ({{96, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_7_BIT** ({{107, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_8_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_8_BIT** ({{11, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_8_BIT** ({{27, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_8_BIT** ({{40, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_8_BIT** ({{42, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D2_8_BIT** ({{44, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_8_BIT** ({{50, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_8_BIT** ({{63, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_8_BIT** ({{74, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_8_BIT** ({{90, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_8_BIT** ({{101, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_8_BIT** ({{117, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_9_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A_9_BIT** ({{11, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T0A_9_BIT** ({{13, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_9_BIT** ({{23, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_9_BIT** ({{39, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_9_BIT** ({{55, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_9_BIT** ({{68, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_9_BIT** ({{70, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SVI_D3_9_BIT** ({{72, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_A_9_BIT** ({{78, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E_9_BIT** ({{91, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_9_BIT** ({{102, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_9_BIT** ({{118, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **IOD_A_10_BIT** ({{7, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA0_10_BIT** ({{11, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT_10_BIT** ({{27, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **M0_10_BIT** ({{38, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF0_10_BIT** ({{54, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **AF1_10_BIT** ({{70, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **E5B_HS_10_BIT** ({{83, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **E1_B_HS_10_BIT** ({{85, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **A_0_G_10_BIT** ({{87, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1_G_10_BIT** ({{103, 12}})

- const std::vector< std::pair< int32_t, int32_t > > **T_0_G_10_BIT** ({{115, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_0_G_10_BIT** ({{123, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_DeltaAred_BIT** ({{7, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_exred_BIT** ({{12, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_eyred_BIT** ({{25, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_Deltai0red_BIT** ({{38, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_Omega0red_BIT** ({{55, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_lambda0red_BIT** ({{78, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_af0red_BIT** ({{101, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **CED_af1red_BIT** ({{123, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **RS_IODNAV_LSBS** ({{15, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_CONSTELLATION_ID_BIT** ({{7, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_SERVICE_LEVEL_ID_BIT** ({{10, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_WN_BIT** ({{13, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_T0_BIT** ({{25, 9}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_MASK_MSB_BIT** ({{34, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_MASK_BIT** ({{35, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_PCONST_BIT** ({{67, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_PSAT_BIT** ({{71, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_URA_BIT** ({{75, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_URE_BIT** ({{79, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_BNOM_BIT** ({{83, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_TVALIDITY_BIT** ({{87, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **ISM_CRC_BIT** ({{97, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **TIME_0_BIT** ({{7, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_0_BIT** ({{97, 12}})
- const std::vector< std::pair< int32_t, int32_t > > **TOW_0_BIT** ({{109, 20}})

Variables

- constexpr double GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS = 2.04

  Page Duration + (Galileo I/NAV Preamble bits)∗(Galileo E5b-I tiered Code Period(seconds)).
- constexpr uint32_t GALILEO_INAV_PAGE_SYMBOLS = 500

  The complete Galileo INAV page length.
- constexpr int32_t GALILEO_INAV_PREAMBLE_LENGTH_BITS = 10
- constexpr int32_t GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS = 250
- constexpr int32_t GALILEO_INAV_PAGE_PART_SYMBOLS = 250

  Each Galileo INAV pages are composed of two parts (even and odd) each of 250 symbols, including preamble. See Galileo ICD 4.3.2.
- constexpr int32_t GALILEO_INAV_PAGE_PART_SECONDS = 1
- constexpr int32_t GALILEO_INAV_PAGE_PART_MS = 1000
- constexpr int32_t GALILEO_INAV_PAGE_SECONDS = 2
- constexpr int32_t GALILEO_INAV_INTERLEAVER_ROWS = 8
- constexpr int32_t GALILEO_INAV_INTERLEAVER_COLS = 30
- constexpr int32_t GALILEO_TELEMETRY_RATE_BITS_SECOND = 250
- constexpr int32_t GALILEO_PAGE_TYPE_BITS = 6
- constexpr int32_t GALILEO_DATA_JK_BITS = 128
- constexpr int32_t GALILEO_DATA_FRAME_BITS = 196
- constexpr int32_t GALILEO_DATA_FRAME_BYTES = 25
- constexpr size_t GALILEO_INAV_EVEN_PAGE_TYPE_BIT = 1
- constexpr size_t GALILEO_INAV_ODD_PAGE_TYPE_BIT = 115
- constexpr int32_t GALILEO_INAV_DUMMY_WORD_TYPE = 63
- constexpr char GALILEO_INAV_PREAMBLE [11] = "0101100000"
- constexpr int32_t T0E_1_LSB = 60
- constexpr double M0_1_LSB = PI_TWO_N31

- constexpr double E_1_LSB = TWO_N33
- constexpr double A_1_LSB_GAL = TWO_N19
- constexpr double OMEGA_0_2_LSB = PI_TWO_N31
- constexpr double I_0_2_LSB = PI_TWO_N31
- constexpr double OMEGA_2_LSB = PI_TWO_N31
- constexpr double I_DOT_2_LSB = PI_TWO_N43
- constexpr double OMEGA_DOT_3_LSB = PI_TWO_N43
- constexpr double DELTA_N_3_LSB = PI_TWO_N43
- constexpr double C_UC_3_LSB = TWO_N29
- constexpr double C_US_3_LSB = TWO_N29
- constexpr double C_RC_3_LSB = TWO_N5
- constexpr double C_RS_3_LSB = TWO_N5
- constexpr double C_IC_4_LSB = TWO_N29
- constexpr double C_IS_4_LSB = TWO_N29
- constexpr int32_t T0C_4_LSB = 60
- constexpr double AF0_4_LSB = TWO_N34
- constexpr double AF1_4_LSB = TWO_N46
- constexpr double AF2_4_LSB = TWO_N59
- constexpr double AI0_5_LSB = TWO_N2
- constexpr double AI1_5_LSB = TWO_N8
- constexpr double AI2_5_LSB = TWO_N15
- constexpr double BGD_E1_E5A_5_LSB = TWO_N32
- constexpr double BGD_E1_E5B_5_LSB = TWO_N32
- constexpr double A0_6_LSB = TWO_N30
- constexpr double A1_6_LSB = TWO_N50
- constexpr int32_t T0T_6_LSB = 3600
- constexpr int32_t T0A_7_LSB = 600
- constexpr double DELTA_A_7_LSB = TWO_N9
- constexpr double E_7_LSB = TWO_N16
- constexpr double OMEGA_7_LSB = TWO_N15
- constexpr double DELTA_I_7_LSB = TWO_N14
- constexpr double OMEGA0_7_LSB = TWO_N15
- constexpr double OMEGA_DOT_7_LSB = TWO_N33
- constexpr double M0_7_LSB = TWO_N15
- constexpr double AF0_8_LSB = TWO_N19
- constexpr double AF1_8_LSB = TWO_N38
- constexpr double DELTA_A_8_LSB = TWO_N9
- constexpr double E_8_LSB = TWO_N16
- constexpr double OMEGA_8_LSB = TWO_N15
- constexpr double DELTA_I_8_LSB = TWO_N14
- constexpr double OMEGA0_8_LSB = TWO_N15
- constexpr double OMEGA_DOT_8_LSB = TWO_N33
- constexpr int32_t T0A_9_LSB = 600
- constexpr double M0_9_LSB = TWO_N15
- constexpr double AF0_9_LSB = TWO_N19
- constexpr double AF1_9_LSB = TWO_N38
- constexpr double DELTA_A_9_LSB = TWO_N9
- constexpr double E_9_LSB = TWO_N16
- constexpr double OMEGA_9_LSB = TWO_N15
- constexpr double DELTA_I_9_LSB = TWO_N14
- constexpr double OMEGA0_10_LSB = TWO_N15
- constexpr double OMEGA_DOT_10_LSB = TWO_N33
- constexpr double M0_10_LSB = TWO_N15
- constexpr double AF0_10_LSB = TWO_N19
- constexpr double AF1_10_LSB = TWO_N38

- constexpr double A_0G_10_LSB = TWO_N35
- constexpr double A_1G_10_LSB = TWO_N51
- constexpr int32_t T_0_G_10_LSB = 3600
- constexpr double CED_DeltaAred_LSB = TWO_P8
- constexpr double CED_exred_LSB = TWO_N22
- constexpr double CED_eyred_LSB = TWO_N22
- constexpr double CED_Deltai0red_LSB = TWO_N22
- constexpr double CED_Omega0red_LSB = TWO_N22
- constexpr double CED_lambda0red_LSB = TWO_N22
- constexpr double CED_af0red_LSB = TWO_N26
- constexpr double CED_af1red_LSB = TWO_N35
- constexpr size_t INAV_RS_SUBVECTOR_LENGTH = 15
- constexpr size_t INAV_RS_PARITY_VECTOR_LENGTH = 60
- constexpr size_t INAV_RS_INFO_VECTOR_LENGTH = 58
- constexpr size_t INAV_RS_BUFFER_LENGTH = 118
- constexpr int32_t BITS_IN_OCTET = 8
- constexpr int32_t FIRST_RS_BIT = 7
- constexpr int32_t FIRST_RS_BIT_AFTER_IODNAV = 17
- constexpr int32_t GALILEO_ISM_CRC_DATA_BITS = 96
- constexpr int32_t GALILEO_ISM_CRC_DATA_BYTES = 12
- constexpr char GALILEO_INAV_PLAIN_SSP1 [9] = "00000100"
- constexpr char GALILEO_INAV_PLAIN_SSP2 [9] = "00101011"
- constexpr char GALILEO_INAV_PLAIN_SSP3 [9] = "00101111"
- constexpr char GALILEO_INAV_ENCODED_SSP1 [17] = "1110100100100101"
- constexpr char GALILEO_INAV_ENCODED_SSP2 [17] = "0110110001001110"
- constexpr char GALILEO_INAV_ENCODED_SSP3 [17] = "1101000000111110"

13.712.1 Detailed Description

Galileo INAV message constants.

Author

Carles Fernandez, 2020. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file Galileo_INAV.h.

13.713 Galileo_INAV.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_INAV.h
00003  * \brief Galileo INAV message constants
00004  * \author Carles Fernandez, 2020. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GALILEO_INAV_H
00019 #define GNSS_SDR_GALILEO_INAV_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include <cstddef>
00023 #include <cstdint>
```

```
00024 #include <utility>
00025 #include <vector>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 // Galileo INAV message structure
00034 constexpr double GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS = 2.04;  //!< Page Duration + (Galileo
      I/NAV Preamble bits)*(Galileo E5b-I tiered Code Period(seconds))
00035 constexpr uint32_t GALILEO_INAV_PAGE_SYMBOLS = 500;                   //!< The complete Galileo INAV
      page length
00036 constexpr int32_t GALILEO_INAV_PREAMBLE_LENGTH_BITS = 10;
00037 constexpr int32_t GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS = 250;
00038 constexpr int32_t GALILEO_INAV_PAGE_PART_SYMBOLS = 250;  //!< Each Galileo INAV pages are composed of
      two parts (even and odd) each of 250 symbols, including preamble. See Galileo ICD 4.3.2
00039 constexpr int32_t GALILEO_INAV_PAGE_PART_SECONDS = 1;    // a page part last 1 sec
00040 constexpr int32_t GALILEO_INAV_PAGE_PART_MS = 1000;      // a page part last 1 sec
00041 constexpr int32_t GALILEO_INAV_PAGE_SECONDS = 2;         // a full page last 2 sec
00042 constexpr int32_t GALILEO_INAV_INTERLEAVER_ROWS = 8;
00043 constexpr int32_t GALILEO_INAV_INTERLEAVER_COLS = 30;
00044 constexpr int32_t GALILEO_TELEMETRY_RATE_BITS_SECOND = 250;  // bps
00045 constexpr int32_t GALILEO_PAGE_TYPE_BITS = 6;
00046 constexpr int32_t GALILEO_DATA_JK_BITS = 128;
00047 constexpr int32_t GALILEO_DATA_FRAME_BITS = 196;
00048 constexpr int32_t GALILEO_DATA_FRAME_BYTES = 25;
00049 constexpr size_t GALILEO_INAV_EVEN_PAGE_TYPE_BIT = 1;
00050 constexpr size_t GALILEO_INAV_ODD_PAGE_TYPE_BIT = 115;
00051 constexpr int32_t GALILEO_INAV_DUMMY_WORD_TYPE = 63;
00052 constexpr char GALILEO_INAV_PREAMBLE[11] = "0101100000";
00053
00054 const std::vector<std::pair<int32_t, int32_t» TYPE({{1, 6}});
00055 const std::vector<std::pair<int32_t, int32_t» PAGE_TYPE_BIT({{1, 6}});
00056
00057 /* Page 1 - Word type 1: Ephemeris (1/4) */
00058 const std::vector<std::pair<int32_t, int32_t» IOD_NAV_1_BIT({{7, 10}});
00059 const std::vector<std::pair<int32_t, int32_t» T0_E_1_BIT({{17, 14}});
00060 constexpr int32_t T0E_1_LSB = 60;
00061 const std::vector<std::pair<int32_t, int32_t» M0_1_BIT({{31, 32}});
00062 constexpr double M0_1_LSB = PI_TWO_N31;
00063 const std::vector<std::pair<int32_t, int32_t» E_1_BIT({{63, 32}});
00064 constexpr double E_1_LSB = TWO_N33;
00065 const std::vector<std::pair<int32_t, int32_t» A_1_BIT({{95, 32}});
00066 constexpr double A_1_LSB_GAL = TWO_N19;
00067 // last two bits are reserved
00068
00069
00070 /* Page 2 - Word type 2: Ephemeris (2/4) */
00071 const std::vector<std::pair<int32_t, int32_t» IOD_NAV_2_BIT({{7, 10}});
00072 const std::vector<std::pair<int32_t, int32_t» OMEGA_0_2_BIT({{17, 32}});
00073 constexpr double OMEGA_0_2_LSB = PI_TWO_N31;
00074 const std::vector<std::pair<int32_t, int32_t» I_0_2_BIT({{49, 32}});
00075 constexpr double I_0_2_LSB = PI_TWO_N31;
00076 const std::vector<std::pair<int32_t, int32_t» OMEGA_2_BIT({{81, 32}});
00077 constexpr double OMEGA_2_LSB = PI_TWO_N31;
00078 const std::vector<std::pair<int32_t, int32_t» I_DOT_2_BIT({{113, 14}});
00079 constexpr double I_DOT_2_LSB = PI_TWO_N43;
00080 // last two bits are reserved
00081
00082 /* Word type 3: Ephemeris (3/4) and SISA */
00083 const std::vector<std::pair<int32_t, int32_t» IOD_NAV_3_BIT({{7, 10}});
00084 const std::vector<std::pair<int32_t, int32_t» OMEGA_DOT_3_BIT({{17, 24}});
00085 constexpr double OMEGA_DOT_3_LSB = PI_TWO_N43;
00086 const std::vector<std::pair<int32_t, int32_t» DELTA_N_3_BIT({{41, 16}});
00087 constexpr double DELTA_N_3_LSB = PI_TWO_N43;
00088 const std::vector<std::pair<int32_t, int32_t» C_UC_3_BIT({{57, 16}});
00089 constexpr double C_UC_3_LSB = TWO_N29;
00090 const std::vector<std::pair<int32_t, int32_t» C_US_3_BIT({{73, 16}});
00091 constexpr double C_US_3_LSB = TWO_N29;
00092 const std::vector<std::pair<int32_t, int32_t» C_RC_3_BIT({{89, 16}});
00093 constexpr double C_RC_3_LSB = TWO_N5;
00094 const std::vector<std::pair<int32_t, int32_t» C_RS_3_BIT({{105, 16}});
00095 constexpr double C_RS_3_LSB = TWO_N5;
00096 const std::vector<std::pair<int32_t, int32_t» SISA_3_BIT({{121, 8}});
00097
00098
00099 /* Word type 4: Ephemeris (4/4) and Clock correction parameters */
00100 const std::vector<std::pair<int32_t, int32_t» IOD_NAV_4_BIT({{7, 10}});
00101 const std::vector<std::pair<int32_t, int32_t» SV_ID_PRN_4_BIT({{17, 6}});
00102 const std::vector<std::pair<int32_t, int32_t» C_IC_4_BIT({{23, 16}});
00103 constexpr double C_IC_4_LSB = TWO_N29;
00104 const std::vector<std::pair<int32_t, int32_t» C_IS_4_BIT({{39, 16}});
00105 constexpr double C_IS_4_LSB = TWO_N29;
00106 const std::vector<std::pair<int32_t, int32_t» T0C_4_BIT({{55, 14}});  //
00107 constexpr int32_t T0C_4_LSB = 60;
```

```
00108 const std::vector<std::pair<int32_t, int32_t» AF0_4_BIT({{69, 31}});  //
00109 constexpr double AF0_4_LSB = TWO_N34;
00110 const std::vector<std::pair<int32_t, int32_t» AF1_4_BIT({{100, 21}});  //
00111 constexpr double AF1_4_LSB = TWO_N46;
00112 const std::vector<std::pair<int32_t, int32_t» AF2_4_BIT({{121, 6}});
00113 constexpr double AF2_4_LSB = TWO_N59;
00114 const std::vector<std::pair<int32_t, int32_t» SPARE_4_BIT({{127, 2}});
00115 // last two bits are reserved
00116
00117 /* Word type 5: Ionospheric correction, BGD, signal health and data validity status and GST */
00118 /* Ionospheric correction */
00119 /* Az */
00120 const std::vector<std::pair<int32_t, int32_t» AI0_5_BIT({{7, 11}});  //
00121 constexpr double AI0_5_LSB = TWO_N2;
00122 const std::vector<std::pair<int32_t, int32_t» AI1_5_BIT({{18, 11}});  //
00123 constexpr double AI1_5_LSB = TWO_N8;
00124 const std::vector<std::pair<int32_t, int32_t» AI2_5_BIT({{29, 14}});  //
00125 constexpr double AI2_5_LSB = TWO_N15;
00126 /* Ionospheric disturbance flag */
00127 const std::vector<std::pair<int32_t, int32_t» REGION1_5_BIT({{43, 1}});      //
00128 const std::vector<std::pair<int32_t, int32_t» REGION2_5_BIT({{44, 1}});      //
00129 const std::vector<std::pair<int32_t, int32_t» REGION3_5_BIT({{45, 1}});      //
00130 const std::vector<std::pair<int32_t, int32_t» REGION4_5_BIT({{46, 1}});      //
00131 const std::vector<std::pair<int32_t, int32_t» REGION5_5_BIT({{47, 1}});      //
00132 const std::vector<std::pair<int32_t, int32_t» BGD_E1_E5A_5_BIT({{48, 10}});  //
00133 constexpr double BGD_E1_E5A_5_LSB = TWO_N32;
00134 const std::vector<std::pair<int32_t, int32_t» BGD_E1_E5B_5_BIT({{58, 10}});  //
00135 constexpr double BGD_E1_E5B_5_LSB = TWO_N32;
00136 const std::vector<std::pair<int32_t, int32_t» E5B_HS_5_BIT({{68, 2}});    //
00137 const std::vector<std::pair<int32_t, int32_t» E1_B_HS_5_BIT({{70, 2}});   //
00138 const std::vector<std::pair<int32_t, int32_t» E5B_DVS_5_BIT({{72, 1}});   //
00139 const std::vector<std::pair<int32_t, int32_t» E1_B_DVS_5_BIT({{73, 1}});  //
00140 /* GST */
00141 const std::vector<std::pair<int32_t, int32_t» WN_5_BIT({{74, 12}});
00142 const std::vector<std::pair<int32_t, int32_t» TOW_5_BIT({{86, 20}});
00143 const std::vector<std::pair<int32_t, int32_t» SPARE_5_BIT({{106, 23}});
00144
00145
00146 /* Page 6 */
00147 const std::vector<std::pair<int32_t, int32_t» A0_6_BIT({{7, 32}});
00148 constexpr double A0_6_LSB = TWO_N30;
00149 const std::vector<std::pair<int32_t, int32_t» A1_6_BIT({{39, 24}});
00150 constexpr double A1_6_LSB = TWO_N50;
00151 const std::vector<std::pair<int32_t, int32_t» DELTA_T_LS_6_BIT({{63, 8}});
00152 const std::vector<std::pair<int32_t, int32_t» T0T_6_BIT({{71, 8}});
00153 constexpr int32_t T0T_6_LSB = 3600;
00154 const std::vector<std::pair<int32_t, int32_t» W_NOT_6_BIT({{79, 8}});
00155 const std::vector<std::pair<int32_t, int32_t» WN_LSF_6_BIT({{87, 8}});
00156 const std::vector<std::pair<int32_t, int32_t» DN_6_BIT({{95, 3}});
00157 const std::vector<std::pair<int32_t, int32_t» DELTA_T_LSF_6_BIT({{98, 8}});
00158 const std::vector<std::pair<int32_t, int32_t» TOW_6_BIT({{106, 20}});
00159
00160
00161 /* Page 7 */
00162 const std::vector<std::pair<int32_t, int32_t» IOD_A_7_BIT({{7, 4}});
00163 const std::vector<std::pair<int32_t, int32_t» WN_A_7_BIT({{11, 2}});
00164 const std::vector<std::pair<int32_t, int32_t» T0A_7_BIT({{13, 10}});
00165 constexpr int32_t T0A_7_LSB = 600;
00166 const std::vector<std::pair<int32_t, int32_t» SVI_D1_7_BIT({{23, 6}});
00167 const std::vector<std::pair<int32_t, int32_t» DELTA_A_7_BIT({{29, 13}});
00168 constexpr double DELTA_A_7_LSB = TWO_N9;
00169 const std::vector<std::pair<int32_t, int32_t» E_7_BIT({{42, 11}});
00170 constexpr double E_7_LSB = TWO_N16;
00171 const std::vector<std::pair<int32_t, int32_t» OMEGA_7_BIT({{53, 16}});
00172 constexpr double OMEGA_7_LSB = TWO_N15;
00173 const std::vector<std::pair<int32_t, int32_t» DELTA_I_7_BIT({{69, 11}});
00174 constexpr double DELTA_I_7_LSB = TWO_N14;
00175 const std::vector<std::pair<int32_t, int32_t» OMEGA0_7_BIT({{80, 16}});
00176 constexpr double OMEGA0_7_LSB = TWO_N15;
00177 const std::vector<std::pair<int32_t, int32_t» OMEGA_DOT_7_BIT({{96, 11}});
00178 constexpr double OMEGA_DOT_7_LSB = TWO_N33;
00179 const std::vector<std::pair<int32_t, int32_t» M0_7_BIT({{107, 16}});
00180 constexpr double M0_7_LSB = TWO_N15;
00181
00182
00183 /* Page 8 */
00184 const std::vector<std::pair<int32_t, int32_t» IOD_A_8_BIT({{7, 4}});
00185 const std::vector<std::pair<int32_t, int32_t» AF0_8_BIT({{11, 16}});
00186 constexpr double AF0_8_LSB = TWO_N19;
00187 const std::vector<std::pair<int32_t, int32_t» AF1_8_BIT({{27, 13}});
00188 constexpr double AF1_8_LSB = TWO_N38;
00189 const std::vector<std::pair<int32_t, int32_t» E5B_HS_8_BIT({{40, 2}});
00190 const std::vector<std::pair<int32_t, int32_t» E1_B_HS_8_BIT({{42, 2}});
00191 const std::vector<std::pair<int32_t, int32_t» SVI_D2_8_BIT({{44, 6}});
00192 const std::vector<std::pair<int32_t, int32_t» DELTA_A_8_BIT({{50, 13}});
00193 constexpr double DELTA_A_8_LSB = TWO_N9;
00194 const std::vector<std::pair<int32_t, int32_t» E_8_BIT({{63, 11}});
```

```
00195 constexpr double E_8_LSB = TWO_N16;
00196 const std::vector<std::pair<int32_t, int32_t» OMEGA_8_BIT({{74, 16}});
00197 constexpr double OMEGA_8_LSB = TWO_N15;
00198 const std::vector<std::pair<int32_t, int32_t» DELTA_I_8_BIT({{90, 11}});
00199 constexpr double DELTA_I_8_LSB = TWO_N14;
00200 const std::vector<std::pair<int32_t, int32_t» OMEGA0_8_BIT({{101, 16}});
00201 constexpr double OMEGA0_8_LSB = TWO_N15;
00202 const std::vector<std::pair<int32_t, int32_t» OMEGA_DOT_8_BIT({{117, 11}});
00203 constexpr double OMEGA_DOT_8_LSB = TWO_N33;
00204
00205
00206 /* Page 9 */
00207 const std::vector<std::pair<int32_t, int32_t» IOD_A_9_BIT({{7, 4}});
00208 const std::vector<std::pair<int32_t, int32_t» WN_A_9_BIT({{11, 2}});
00209 const std::vector<std::pair<int32_t, int32_t» T0A_9_BIT({{13, 10}});
00210 constexpr int32_t T0A_9_LSB = 600;
00211 const std::vector<std::pair<int32_t, int32_t» M0_9_BIT({{23, 16}});
00212 constexpr double M0_9_LSB = TWO_N15;
00213 const std::vector<std::pair<int32_t, int32_t» AF0_9_BIT({{39, 16}});
00214 constexpr double AF0_9_LSB = TWO_N19;
00215 const std::vector<std::pair<int32_t, int32_t» AF1_9_BIT({{55, 13}});
00216 constexpr double AF1_9_LSB = TWO_N38;
00217 const std::vector<std::pair<int32_t, int32_t» E5B_HS_9_BIT({{68, 2}});
00218 const std::vector<std::pair<int32_t, int32_t» E1_B_HS_9_BIT({{70, 2}});
00219 const std::vector<std::pair<int32_t, int32_t» SVI_D3_9_BIT({{72, 6}});
00220 const std::vector<std::pair<int32_t, int32_t» DELTA_A_9_BIT({{78, 13}});
00221 constexpr double DELTA_A_9_LSB = TWO_N9;
00222 const std::vector<std::pair<int32_t, int32_t» E_9_BIT({{91, 11}});
00223 constexpr double E_9_LSB = TWO_N16;
00224 const std::vector<std::pair<int32_t, int32_t» OMEGA_9_BIT({{102, 16}});
00225 constexpr double OMEGA_9_LSB = TWO_N15;
00226 const std::vector<std::pair<int32_t, int32_t» DELTA_I_9_BIT({{118, 11}});
00227 constexpr double DELTA_I_9_LSB = TWO_N14;
00228
00229
00230 /* Page 10 */
00231 const std::vector<std::pair<int32_t, int32_t» IOD_A_10_BIT({{7, 4}});
00232 const std::vector<std::pair<int32_t, int32_t» OMEGA0_10_BIT({{11, 16}});
00233 constexpr double OMEGA0_10_LSB = TWO_N15;
00234 const std::vector<std::pair<int32_t, int32_t» OMEGA_DOT_10_BIT({{27, 11}});
00235 constexpr double OMEGA_DOT_10_LSB = TWO_N33;
00236 const std::vector<std::pair<int32_t, int32_t» M0_10_BIT({{38, 16}});
00237 constexpr double M0_10_LSB = TWO_N15;
00238 const std::vector<std::pair<int32_t, int32_t» AF0_10_BIT({{54, 16}});
00239 constexpr double AF0_10_LSB = TWO_N19;
00240 const std::vector<std::pair<int32_t, int32_t» AF1_10_BIT({{70, 13}});
00241 constexpr double AF1_10_LSB = TWO_N38;
00242 const std::vector<std::pair<int32_t, int32_t» E5B_HS_10_BIT({{83, 2}});
00243 const std::vector<std::pair<int32_t, int32_t» E1_B_HS_10_BIT({{85, 2}});
00244 const std::vector<std::pair<int32_t, int32_t» A_0_G_10_BIT({{87, 16}});
00245 constexpr double A_0G_10_LSB = TWO_N35;
00246 const std::vector<std::pair<int32_t, int32_t» A_1_G_10_BIT({{103, 12}});
00247 constexpr double A_1G_10_LSB = TWO_N51;
00248 const std::vector<std::pair<int32_t, int32_t» T_0_G_10_BIT({{115, 8}});
00249 constexpr int32_t T_0_G_10_LSB = 3600;
00250 const std::vector<std::pair<int32_t, int32_t» WN_0_G_10_BIT({{123, 6}});
00251
00252 /* Page 16 */
00253 constexpr double CED_DeltaAred_LSB = TWO_P8;
00254 const std::vector<std::pair<int32_t, int32_t» CED_DeltaAred_BIT({{7, 5}});
00255 constexpr double CED_exred_LSB = TWO_N22;
00256 const std::vector<std::pair<int32_t, int32_t» CED_exred_BIT({{12, 13}});
00257 constexpr double CED_eyred_LSB = TWO_N22;
00258 const std::vector<std::pair<int32_t, int32_t» CED_eyred_BIT({{25, 13}});
00259 constexpr double CED_Deltai0red_LSB = TWO_N22;
00260 const std::vector<std::pair<int32_t, int32_t» CED_Deltai0red_BIT({{38, 17}});
00261 constexpr double CED_Omega0red_LSB = TWO_N22;
00262 const std::vector<std::pair<int32_t, int32_t» CED_Omega0red_BIT({{55, 23}});
00263 constexpr double CED_lambda0red_LSB = TWO_N22;
00264 const std::vector<std::pair<int32_t, int32_t» CED_lambda0red_BIT({{78, 23}});
00265 constexpr double CED_af0red_LSB = TWO_N26;
00266 const std::vector<std::pair<int32_t, int32_t» CED_af0red_BIT({{101, 22}});
00267 constexpr double CED_af1red_LSB = TWO_N35;
00268 const std::vector<std::pair<int32_t, int32_t» CED_af1red_BIT({{123, 6}});
00269
00270 /* Pages 17, 18, 19, 20 */
00271 const std::vector<std::pair<int32_t, int32_t» RS_IODNAV_LSBS({{15, 2}});
00272 constexpr size_t INAV_RS_SUBVECTOR_LENGTH = 15;
00273 constexpr size_t INAV_RS_PARITY_VECTOR_LENGTH = 60;
00274 constexpr size_t INAV_RS_INFO_VECTOR_LENGTH = 58;
00275 constexpr size_t INAV_RS_BUFFER_LENGTH = 118;
00276 constexpr int32_t BITS_IN_OCTET = 8;
00277 constexpr int32_t FIRST_RS_BIT = 7;
00278 constexpr int32_t FIRST_RS_BIT_AFTER_IODNAV = 17;
00279
00280 /* Page 22 */
00281 const std::vector<std::pair<int32_t, int32_t» ISM_CONSTELLATION_ID_BIT({{7, 3}});
```

```
00282 const std::vector<std::pair<int32_t, int32_t» ISM_SERVICE_LEVEL_ID_BIT({{10, 3}});
00283 const std::vector<std::pair<int32_t, int32_t» ISM_WN_BIT({{13, 12}});
00284 const std::vector<std::pair<int32_t, int32_t» ISM_T0_BIT({{25, 9}});
00285 const std::vector<std::pair<int32_t, int32_t» ISM_MASK_MSB_BIT({{34, 1}});
00286 const std::vector<std::pair<int32_t, int32_t» ISM_MASK_BIT({{35, 32}});
00287 const std::vector<std::pair<int32_t, int32_t» ISM_PCONST_BIT({{67, 4}});
00288 const std::vector<std::pair<int32_t, int32_t» ISM_PSAT_BIT({{71, 4}});
00289 const std::vector<std::pair<int32_t, int32_t» ISM_URA_BIT({{75, 4}});
00290 const std::vector<std::pair<int32_t, int32_t» ISM_URE_BIT({{79, 4}});
00291 const std::vector<std::pair<int32_t, int32_t» ISM_BNOM_BIT({{83, 4}});
00292 const std::vector<std::pair<int32_t, int32_t» ISM_TVALIDITY_BIT({{87, 4}});
00293 const std::vector<std::pair<int32_t, int32_t» ISM_CRC_BIT({{97, 32}});
00294 constexpr int32_t GALILEO_ISM_CRC_DATA_BITS = 96;
00295 constexpr int32_t GALILEO_ISM_CRC_DATA_BYTES = 12;
00296
00297 /* Page 0 */
00298 const std::vector<std::pair<int32_t, int32_t» TIME_0_BIT({{7, 2}});
00299 const std::vector<std::pair<int32_t, int32_t» WN_0_BIT({{97, 12}});
00300 const std::vector<std::pair<int32_t, int32_t» TOW_0_BIT({{109, 20}});
00301
00302 /* Secondary Synchronization Patterns */
00303 constexpr char GALILEO_INAV_PLAIN_SSP1[9] = "00000100";
00304 constexpr char GALILEO_INAV_PLAIN_SSP2[9] = "00101011";
00305 constexpr char GALILEO_INAV_PLAIN_SSP3[9] = "00101111";
00306 constexpr char GALILEO_INAV_ENCODED_SSP1[17] = "1110100100100101";
00307 constexpr char GALILEO_INAV_ENCODED_SSP2[17] = "0110110001001110";
00308 constexpr char GALILEO_INAV_ENCODED_SSP3[17] = "1101000000111110";
00309
00310 /** \} */
00311 /** \} */
00312 #endif  // GNSS_SDR_GALILEO_INAV_H
```

13.714 galileo_inav_message.h File Reference

Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 2.0 (Jan. 2021).

```
#include "Galileo_INAV.h"
#include "galileo_almanac_helper.h"
#include "galileo_ephemeris.h"
#include "galileo_iono.h"
#include "galileo_ism.h"
#include "galileo_reduced_ced.h"
#include "galileo_utc_model.h"
#include "gnss_sdr_make_unique.h"
#include <array>
#include <bitset>
#include <cstdint>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class OSNMA_msg

  This class fills the OSNMA_msg structure with the data received from the telemetry blocks.

- class Galileo_Inav_Message

  This class handles the Galileo I/NAV Data message, as described in the Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan. 2021). See https://www.gsc-europa.eu/↩ sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf.

13.714.1 Detailed Description

Implementation of a Galileo I/NAV Data message as described in Galileo OS SIS ICD Issue 2.0 (Jan. 2021).

Author

Mara Branzanti 2013. mara.branzanti(at)gmail.com

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_inav_message.h.

13.715 galileo_inav_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_inav_message.h
00003  * \brief  Implementation of a Galileo I/NAV Data message
00004  *         as described in Galileo OS SIS ICD Issue 2.0 (Jan. 2021)
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GALILEO_INAV_MESSAGE_H
00020 #define GNSS_SDR_GALILEO_INAV_MESSAGE_H
00021
00022 #include "Galileo_INAV.h"
00023 #include "galileo_almanac_helper.h"
00024 #include "galileo_ephemeris.h"
00025 #include "galileo_iono.h"
00026 #include "galileo_ism.h"
00027 #include "galileo_reduced_ced.h"
00028 #include "galileo_utc_model.h"
00029 #include "gnss_sdr_make_unique.h"  // for std::unique_ptr in C++11
00030 #include <array>
00031 #include <bitset>
00032 #include <cstdint>
00033 #include <memory>
00034 #include <string>
00035 #include <utility>
00036 #include <vector>
00037
00038 class ReedSolomon;  // Forward declaration of the ReedSolomon class
00039
00040 /** \addtogroup Core
00041  * \{ */
00042 /** \addtogroup System_Parameters
00043  * \{ */
00044 /*!
00045  * \brief This class fills the OSNMA_msg structure with the data received from the telemetry blocks.
00046  */
00047 class OSNMA_msg
00048 {
00049 public:
00050     OSNMA_msg() = default;
00051     std::array<uint32_t, 15> mack{};
00052     std::array<uint8_t, 15> hkroot{};
00053     std::array<uint8_t, 15> page_valid{};
00054     bool page_validity_available{false};
00055     uint32_t PRN{};      // PRN_a authentication data PRN
00056     uint32_t WN_sf0{};   // Week number at the start of OSNMA subframe
00057     uint32_t TOW_sf0{};  // TOW at the start of OSNMA subframe
00058 };
00059
00060 /*!
00061  * \brief This class handles the Galileo I/NAV Data message, as described in the
00062  * Galileo Open Service Signal in Space Interface Control Document (OS SIS ICD), Issue 2.0 (Jan.
      2021).
00063  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
00064  */
00065 class Galileo_Inav_Message
00066 {
00067 public:
00068     Galileo_Inav_Message();
00069
```

```
00070     ~Galileo_Inav_Message();
00071
00072     /*
00073      * \brief Takes in input a page (Odd or Even) of 120 bit, split it according ICD 4.3.2.3 and join
      Data_k with Data_j
00074      */
00075     void split_page(std::string page_string, int32_t flag_even_word);
00076
00077     /*
00078      * \brief Returns true if new Ephemeris has arrived. The flag is set to false when the function is
      executed
00079      */
00080     bool have_new_ephemeris();
00081
00082     /*
00083      * \brief Returns true if new Iono model has arrived. The flag is set to false when the function
      is executed
00084      */
00085     bool have_new_iono_and_GST();
00086
00087     /*
00088      * \brief Returns true if new UTC model has arrived. The flag is set to false when the function is
      executed
00089      */
00090     bool have_new_utc_model();
00091
00092     /*
00093      * \brief Returns true if new UTC model has arrived. The flag is set to false when the function is
      executed
00094      */
00095     bool have_new_almanac();
00096
00097     /*
00098      * \brief Returns true if new Reduced CED parameters have arrived. The flag is set to false when
      the function is executed
00099      */
00100     bool have_new_reduced_ced();
00101
00102     /*
00103      * \brief Returns true if new ISM data have arrived. The flag is set to false when the function is
      executed
00104      */
00105     bool have_new_ism();
00106
00107     /*
00108      * \brief Returns true if new NMA data have arrived. The flag is set to false when the function is
      executed
00109      */
00110     bool have_new_nma();
00111
00112     /*
00113      * \brief Returns a Galileo_Ephemeris object filled with the latest navigation data received
00114      */
00115     Galileo_Ephemeris get_ephemeris() const;
00116
00117     /*
00118      * \brief Returns a Galileo_Iono object filled with the latest navigation data received
00119      */
00120     Galileo_Iono get_iono() const;
00121
00122     /*
00123      * \brief Returns a Galileo_Utc_Model object filled with the latest navigation data received
00124      */
00125     Galileo_Utc_Model get_utc_model() const;
00126
00127     /*
00128      * \brief Returns a Galileo_Almanac_Helper object filled with the latest navigation data received
00129      */
00130     Galileo_Almanac_Helper get_almanac() const;
00131
00132     /*
00133      * \brief Returns a Galileo_Reduced_CED object filled with the latest Word 16 and Word 5 data
      received
00134      */
00135     Galileo_Reduced_CED get_reduced_ced() const;
00136
00137     /*
00138      * \brief Returns a Galileo_ISMs object filled with the latest ISM data received
00139      */
00140     Galileo_ISM get_galileo_ism() const;
00141
00142     /*
00143      * \brief Returns an OSNMA_msg object filled with the latest NMA message received. Resets msg
      buffer.
00144      */
00145     OSNMA_msg get_osnma_msg();
00146
```

```
00147     /*
00148      * @brief Retrieves the OSNMA ADKD 4 NAV bits. Resets the string.
00149      */
00150     std::string get_osnma_adkd_4_nav_bits();
00151
00152     /*
00153      * @brief Resets the OSNMA ADKD 4 NAV bits.
00154      */
00155     void reset_osnma_nav_bits_adkd4();
00156
00157     /*
00158      * @brief Retrieves the OSNMA ADKD 0/12 NAV bits. Resets the string.
00159      */
00160     std::string get_osnma_adkd_0_12_nav_bits();
00161
00162     /*
00163      * @brief Resets the OSNMA ADKD 0/12 NAV bits.
00164      */
00165     void reset_osnma_nav_bits_adkd0_12();
00166
00167     inline bool get_flag_CRC_test() const
00168     {
00169         return flag_CRC_test;
00170     }
00171
00172     inline bool is_alert_page() const
00173     {
00174         return flag_alert_page;
00175     }
00176
00177     inline bool get_flag_TOW_set() const
00178     {
00179         return flag_TOW_set;
00180     }
00181
00182     inline void set_flag_TOW_set(bool flag_tow)
00183     {
00184         flag_TOW_set = flag_tow;
00185     }
00186
00187     inline int32_t get_Galileo_week() const
00188     {
00189         return WN_0;
00190     }
00191
00192     inline int32_t get_TOW5() const
00193     {
00194         return TOW_5;
00195     }
00196
00197     inline int32_t get_TOW6() const
00198     {
00199         return TOW_6;
00200     }
00201
00202     inline bool is_TOW5_set() const
00203     {
00204         return flag_TOW_5;
00205     }
00206
00207     inline void set_TOW5_flag(bool flag_tow5)
00208     {
00209         flag_TOW_5 = flag_tow5;
00210     }
00211
00212     inline bool is_TOW6_set() const
00213     {
00214         return flag_TOW_6;
00215     }
00216
00217     inline void set_TOW6_flag(bool flag_tow6)
00218     {
00219         flag_TOW_6 = flag_tow6;
00220     }
00221
00222     inline int32_t get_TOW0() const
00223     {
00224         return TOW_0;
00225     }
00226
00227     inline bool is_TOW0_set() const
00228     {
00229         return flag_TOW_0;
00230     }
00231
00232     inline void set_TOW0_flag(bool flag_tow0)
00233     {
```

```
00234         flag_TOW_0 = flag_tow0;
00235     }
00236
00237     inline bool get_flag_GGTO() const
00238     {
00239         return (flag_GGTO_1 == true && flag_GGTO_2 == true && flag_GGTO_3 == true && flag_GGTO_4 ==
      true && flag_GGTO_valid == true);
00240     }
00241
00242     inline double get_A0G() const
00243     {
00244         return A_0G_10;
00245     }
00246
00247     inline double get_A1G() const
00248     {
00249         return A_1G_10;
00250     }
00251
00252     inline double get_t0G() const
00253     {
00254         return t_0G_10;
00255     }
00256
00257     inline double get_WN0G() const
00258     {
00259         return WN_0G_10;
00260     }
00261
00262     /*
00263      * \brief Initialize PRN field so we do not need to wait for page 4.
00264      */
00265     inline void init_PRN(uint32_t prn)
00266     {
00267         SV_ID_PRN_4 = prn;
00268         flag_CED = false;
00269         flag_rs_recovered_ephemeris_pending = false;
00270         word_5_generation = 0U;
00271         reduced_ced_word_5_generation = 0U;
00272         rs_recovery_word_5_generation = 0U;
00273         nma_msg.PRN = prn;
00274         nma_msg.mack = std::array<uint32_t, 15>{};
00275         nma_msg.hkroot = std::array<uint8_t, 15>{};
00276         nma_msg.page_valid = std::array<uint8_t, 15>{};
00277         nma_msg.page_validity_available = false;
00278         page_position_in_inav_subframe = 255;
00279         nma_position_filled = std::array<int8_t, 15>{};
00280     }
00281
00282     /*
00283      * \brief Enable Reed-Solomon in Galileo E1B
00284      */
00285     inline void enable_reed_solomon()
00286     {
00287         enable_rs = true;
00288     }
00289
00290 private:
00291     bool CRC_test(const std::bitset<GALILEO_DATA_FRAME_BITS>& bits, uint32_t checksum) const;
00292     bool read_navigation_bool(const std::bitset<GALILEO_DATA_JK_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00293     uint64_t read_navigation_unsigned(const std::bitset<GALILEO_DATA_JK_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00294     uint64_t read_page_type_unsigned(const std::bitset<GALILEO_PAGE_TYPE_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00295     int64_t read_navigation_signed(const std::bitset<GALILEO_DATA_JK_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00296     uint8_t read_octet_unsigned(const std::bitset<GALILEO_DATA_JK_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00297     void read_page_1(const std::bitset<GALILEO_DATA_JK_BITS>& data_bits);
00298     void read_page_2(const std::bitset<GALILEO_DATA_JK_BITS>& data_bits);
00299     void read_page_3(const std::bitset<GALILEO_DATA_JK_BITS>& data_bits);
00300     void read_page_4(const std::bitset<GALILEO_DATA_JK_BITS>& data_bits);
00301     std::bitset<GALILEO_DATA_JK_BITS> regenerate_page_1(const std::vector<uint8_t>& decoded) const;
00302     std::bitset<GALILEO_DATA_JK_BITS> regenerate_page_2(const std::vector<uint8_t>& decoded) const;
00303     std::bitset<GALILEO_DATA_JK_BITS> regenerate_page_3(const std::vector<uint8_t>& decoded) const;
00304     std::bitset<GALILEO_DATA_JK_BITS> regenerate_page_4(const std::vector<uint8_t>& decoded) const;
00305
00306     Galileo_ISM gal_ism{};
00307     std::string page_Even{};
00308
00309     std::vector<uint8_t> rs_buffer;   // Reed-Solomon buffer
00310     std::unique_ptr<ReedSolomon> rs;  // The Reed-Solomon decoder
00311     std::vector<int> inav_rs_pages;   // Pages 1,2,3,4,17,18,19,20. Holds 1 if the page has arrived, 0
      otherwise.
00312
00313     int32_t page_jk_decoder(const char* data_jk);
```

```
00314     int32_t IOD_ephemeris{};
00315
00316     // Word type 1: Ephemeris (1/4)
00317     int32_t IOD_nav_1{};  // IOD_nav page 1
00318     int32_t t0e_1{};      // Ephemeris reference time [s]
00319     double M0_1{};        // Mean anomaly at reference time [rad]
00320     double e_1{};         // Eccentricity
00321     double A_1{};         // Square root of the semi-major axis [meters^1/2]
00322
00323     // Word type 2: Ephemeris (2/4)
00324     int32_t IOD_nav_2{};  // IOD_nav page 2
00325     double OMEGA_0_2{};   // Longitude of ascending node of orbital plane at weekly epoch [rad]
00326     double i_0_2{};       // Inclination angle at reference time  [rad]
00327     double omega_2{};     // Argument of perigee [rad]
00328     double iDot_2{};      // Rate of inclination angle [rad/sec]
00329
00330     // Word type 3: Ephemeris (3/4) and SISA
00331     int32_t IOD_nav_3{};
00332     int32_t SISA_3{};
00333     double OMEGA_dot_3{};  // Rate of right ascension [rad/sec]
00334     double delta_n_3{};    // Mean motion difference from computed value  [rad/sec]
00335     double C_uc_3{};       // Amplitude of the cosine harmonic correction term to the argument of
      latitude [radians]
00336     double C_us_3{};       // Amplitude of the sine harmonic correction term to the argument of
      latitude [radians]
00337     double C_rc_3{};       // Amplitude of the cosine harmonic correction term to the orbit radius
      [meters]
00338     double C_rs_3{};       // Amplitude of the sine harmonic correction term to the orbit radius
      [meters]
00339
00340     // Word type 4: Ephemeris (4/4) and Clock correction parameters*/
00341     int32_t IOD_nav_4{};    //
00342     int32_t SV_ID_PRN_4{};  //
00343     double C_ic_4{};        // Amplitude of the cosine harmonic correction term to the angle of
      inclination [radians]
00344     double C_is_4{};        // Amplitude of the sine harmonic correction term to the angle of
      inclination [radians]
00345
00346     // Clock correction parameters
00347     int32_t t0c_4{};  // Clock correction data reference Time of Week [sec]
00348     double af0_4{};   // SV clock bias correction coefficient [s]
00349     double af1_4{};   // SV clock drift correction coefficient [s/s]
00350     double af2_4{};   // clock drift rate correction coefficient [s/s^2]
00351     double spare_4{};
00352
00353     //  Word type 5: Ionospheric correction, BGD, signal health and data validity status and GST*/
00354     // Ionospheric correction
00355     double ai0_5{};        // Effective Ionisation Level 1st order parameter [sfu]
00356     double ai1_5{};        // Effective Ionisation Level 2st order parameter [sfu/degree]
00357     double ai2_5{};        // Effective Ionisation Level 3st order parameter [sfu/degree]
00358     double BGD_E1E5a_5{};  // E1-E5a Broadcast Group Delay [s]
00359     double BGD_E1E5b_5{};  // E1-E5b Broadcast Group Delay [s]
00360     int32_t E5b_HS_5{};    // E5b Signal Health Status
00361     int32_t E1B_HS_5{};    // E1B Signal Health Status
00362
00363     // Ionospheric disturbance flag
00364     bool Region1_flag_5{};  // Ionospheric Disturbance Flag for region 1
00365     bool Region2_flag_5{};  // Ionospheric Disturbance Flag for region 2
00366     bool Region3_flag_5{};  // Ionospheric Disturbance Flag for region 3
00367     bool Region4_flag_5{};  // Ionospheric Disturbance Flag for region 4
00368     bool Region5_flag_5{};  // Ionospheric Disturbance Flag for region 5
00369     bool E5b_DVS_5{};       // E5b Data Validity Status
00370     bool E1B_DVS_5{};       // E1B Data Validity Status
00371
00372     // GST
00373     int32_t WN_5{};
00374     int32_t TOW_5{};
00375     double spare_5{};
00376
00377     // Word type 6: GST-UTC conversion parameters
00378     double A0_6{};
00379     double A1_6{};
00380     int32_t Delta_tLS_6{};
00381     int32_t t0t_6{};
00382     int32_t WNot_6{};
00383     int32_t WN_LSF_6{};
00384     int32_t DN_6{};
00385     int32_t Delta_tLSF_6{};
00386     int32_t TOW_6{};
00387
00388     // Word type 7: Almanac for SVID1 (1/2), almanac reference time and almanac reference week number
00389     int32_t IOD_a_7{};
00390     int32_t WN_a_7{};
00391     int32_t t0a_7{};
00392     int32_t SVID1_7{};
00393     double DELTA_A_7{};
00394     double e_7{};
```

```
00395     double omega_7{};
00396     double delta_i_7{};
00397     double Omega0_7{};
00398     double Omega_dot_7{};
00399     double M0_7{};
00400
00401     // Word type 8: Almanac for SVID1 (2/2) and SVID2 (1/2)
00402     int32_t IOD_a_8{};
00403     int32_t E5b_HS_8{};
00404     int32_t E1B_HS_8{};
00405     int32_t SVID2_8{};
00406     double af0_8{};
00407     double af1_8{};
00408     double DELTA_A_8{};
00409     double e_8{};
00410     double omega_8{};
00411     double delta_i_8{};
00412     double Omega0_8{};
00413     double Omega_dot_8{};
00414
00415     // Word type 9: Almanac for SVID2 (2/2) and SVID3 (1/2)
00416     int32_t IOD_a_9{};
00417     int32_t WN_a_9{};
00418     int32_t t0a_9{};
00419     double M0_9{};
00420     double af0_9{};
00421     double af1_9{};
00422     int32_t E5b_HS_9{};
00423     int32_t E1B_HS_9{};
00424     int32_t SVID3_9{};
00425     double DELTA_A_9{};
00426     double e_9{};
00427     double omega_9{};
00428     double delta_i_9{};
00429
00430     //  Word type 10: Almanac for SVID3 (2/2) and GST-GPS conversion parameters
00431     int32_t IOD_a_10{};
00432     double Omega0_10{};
00433     double Omega_dot_10{};
00434     double M0_10{};
00435     double af0_10{};
00436     double af1_10{};
00437     int32_t E5b_HS_10{};
00438     int32_t E1B_HS_10{};
00439
00440     // GST-GPS conversion
00441     double A_0G_10{};    // Constant term of the offset Delta t systems
00442     double A_1G_10{};    // Rate of change of the offset Delta t systems
00443     int32_t t_0G_10{};   // Reference time for Galileo/GPS Time Offset (GGTO) data
00444     int32_t WN_0G_10{};  // Week Number of Galileo/GPS Time Offset (GGTO) reference
00445
00446     // Word type 0: I/NAV Spare Word
00447     int32_t Time_0{};
00448     int32_t WN_0{};
00449     int32_t TOW_0{};
00450
00451     // Word type 16: Reduced Clock and Ephemeris Data (CED) parameters
00452     double ced_DeltaAred{};
00453     double ced_exred{};
00454     double ced_eyred{};
00455     double ced_Deltai0red{};
00456     double ced_Omega0red{};
00457     double ced_lambda0red{};
00458     double ced_af0red{};
00459     double ced_af1red{};
00460
00461     double Galileo_satClkDrift{};
00462
00463     int32_t current_IODnav{};
00464
00465     // OSNMA
00466     uint32_t mack_sis{};
00467     uint8_t hkroot_sis{};
00468     uint8_t page_position_in_inav_subframe{255};
00469     std::array<int8_t, 15> nma_position_filled{};
00470     OSNMA_msg nma_msg{};
00471     std::string nav_bits_adkd_4{};
00472     std::string nav_bits_word_6{};
00473     std::string nav_bits_word_10{};
00474     std::string nav_bits_adkd_0_12{};
00475     std::string nav_bits_word_1{};
00476     std::string nav_bits_word_2{};
00477     std::string nav_bits_word_3{};
00478     std::string nav_bits_word_4{};
00479     std::string nav_bits_word_5{};
00480
00481     uint8_t IODnav_LSB17{};
```

```
00482     uint8_t IODnav_LSB18{};
00483     uint8_t IODnav_LSB19{};
00484     uint8_t IODnav_LSB20{};
00485
00486     uint64_t word_5_generation{};
00487     uint64_t reduced_ced_word_5_generation{};
00488     uint64_t rs_recovery_word_5_generation{};
00489
00490     uint8_t ism_constellation_id{};
00491     uint8_t ism_service_level_id{};
00492
00493     bool flag_CRC_test{};
00494     bool flag_alert_page{};     // Flag indicating that the last complete page (even + odd) was an
      alert page (Page Type = 1), whose CRC cannot be verified on a single frequency
00495     bool flag_all_ephemeris{};  // Flag indicating that all words containing ephemeris have been
      received
00496     bool flag_ephemeris_1{};    // Flag indicating that ephemeris 1/4 (word 1) have been received
00497     bool flag_ephemeris_2{};    // Flag indicating that ephemeris 2/4 (word 2) have been received
00498     bool flag_ephemeris_3{};    // Flag indicating that ephemeris 3/4 (word 3) have been received
00499     bool flag_ephemeris_4{};    // Flag indicating that ephemeris 4/4 (word 4) have been received
00500
00501     bool flag_iono_and_GST{};  // Flag indicating that ionospheric and GST parameters (word 5) have
      been received
00502     bool flag_iono_model_valid{};
00503     bool flag_TOW_5{};
00504     bool flag_TOW_6{};
00505     bool flag_TOW_0{};
00506     bool flag_TOW_set{};          // it is true when page 5 or page 6 arrives
00507     bool flag_utc_model_valid{};  // Flag indicating that UTC model parameters (word 6) have been
      received
00508     bool flag_new_utc_model{};    // Flag indicating that a newly decoded UTC model is pending
      publication
00509
00510     bool flag_all_almanac{};  // Flag indicating that all Almanac data have been received
00511     bool flag_almanac_1{};    // Flag indicating that almanac 1/4 (word 7) have been received
00512     bool flag_almanac_2{};    // Flag indicating that almanac 2/4 (word 8) have been received
00513     bool flag_almanac_3{};    // Flag indicating that almanac 3/4 (word 9) have been received
00514     bool flag_almanac_4{};    // Flag indicating that almanac 4/4 (word 10) have been received
00515
00516     bool flag_GGTO_1{};
00517     bool flag_GGTO_2{};
00518     bool flag_GGTO_3{};
00519     bool flag_GGTO_4{};
00520     bool flag_GGTO_valid{};
00521
00522     bool flag_CED{};
00523     bool flag_rs_recovered_ephemeris_pending{};
00524     bool enable_rs{};
00525     bool have_ISM{};
00526     bool current_IODnav_valid{};  // IOD_nav can be zero, so track initialization separately.
00527 };
00528
00529
00530 /** \} */
00531 /** \} */
00532 #endif  // GNSS_SDR_GALILEO_INAV_MESSAGE_H
```

13.716 galileo_iono.h File Reference

Interface of a Galileo Ionospheric Model storage.

```
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <cstdint>
```

Classes

- class Galileo_Iono

  This class is a storage for the GALILEO IONOSPHERIC data as described in Galileo ICD paragraph 5.1.6.

13.716.1 Detailed Description

Interface of a Galileo Ionospheric Model storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

Mara Branzanti 2013. mara.branzanti(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_iono.h.

13.717 galileo_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_iono.h
00003  * \brief  Interface of a Galileo Ionospheric Model storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  * \author Mara Branzanti 2013. mara.branzanti(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GALILEO_IONO_H
00020 #define GNSS_SDR_GALILEO_IONO_H
00021
00022 #include <boost/serialization/nvp.hpp>
00023 #include <boost/serialization/version.hpp>
00024 #include <cstdint>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class is a storage for the GALILEO IONOSPHERIC data as described
00034  * in Galileo ICD paragraph 5.1.6
00035  *
00036  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
00037  */
00038 class Galileo_Iono
00039 {
00040 public:
00041     /*!
00042      * Default constructor
00043      */
00044     Galileo_Iono() = default;
00045
00046     // Ionospheric correction
00047     double ai0{};  //!< Effective Ionisation Level 1st order parameter [sfu]
00048     double ai1{};  //!< Effective Ionisation Level 2st order parameter [sfu/degree]
00049     double ai2{};  //!< Effective Ionisation Level 3st order parameter [sfu/degree]
00050
00051     // from page 5 (UTC) to have a timestamp
00052     int32_t tow{};  //!< UTC data reference Time of Week [s]
00053     int32_t WN{};   //!< UTC data reference Week number [week]
00054
00055     // Ionospheric disturbance flag
00056     bool Region1_flag{};  //!< Ionospheric Disturbance Flag for region 1
00057     bool Region2_flag{};  //!< Ionospheric Disturbance Flag for region 2
00058     bool Region3_flag{};  //!< Ionospheric Disturbance Flag for region 3
00059     bool Region4_flag{};  //!< Ionospheric Disturbance Flag for region 4
00060     bool Region5_flag{};  //!< Ionospheric Disturbance Flag for region 5
00061     bool valid{};         //!< True after a complete ionospheric model has been decoded
00062
00063     template <class Archive>
00064
00065     /*!
00066      * \brief Serialize is a boost standard method to be called by the boost XML serialization.
00067      Here is used to save the iono data on disk file.
00068      */
00069     inline void serialize(Archive& archive, const unsigned int version)
00070     {
```

```
00071         if (version)
00072             {
00073             };
00074         archive& BOOST_SERIALIZATION_NVP(ai0);
00075         archive& BOOST_SERIALIZATION_NVP(ai1);
00076         archive& BOOST_SERIALIZATION_NVP(ai2);
00077         archive& BOOST_SERIALIZATION_NVP(tow);
00078         archive& BOOST_SERIALIZATION_NVP(WN);
00079         archive& BOOST_SERIALIZATION_NVP(Region1_flag);
00080         archive& BOOST_SERIALIZATION_NVP(Region2_flag);
00081         archive& BOOST_SERIALIZATION_NVP(Region3_flag);
00082         archive& BOOST_SERIALIZATION_NVP(Region4_flag);
00083         archive& BOOST_SERIALIZATION_NVP(Region5_flag);
00084         if (version > 0)
00085             {
00086                 archive& BOOST_SERIALIZATION_NVP(valid);
00087             }
00088     }
00089 };
00090
00091
00092 BOOST_CLASS_VERSION(Galileo_Iono, 1)
00093
00094
00095 /** \} */
00096 /** \} */
00097 #endif  // GNSS_SDR_GALILEO_IONO_H
```

13.718 galileo_ism.h File Reference

Interface of a Galileo Integrity Support Message.

```
#include "Galileo_INAV.h"
#include <boost/crc.hpp>
#include <bitset>
#include <cstdint>
#include <unordered_map>
#include <vector>
```

Classes

- class Galileo_ISM

  This class is a storage for the GALILEO Integrity Support Message as described in Galileo ICD paragraph 5.2.

13.718.1 Detailed Description

Interface of a Galileo Integrity Support Message.

Author

Carles Fernandez, 2024. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2024 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_ism.h.

13.719 galileo_ism.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_ism.h
00003  * \brief  Interface of a Galileo Integrity Support Message
00004  * \author Carles Fernandez, 2024. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2024  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
```

```
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GALILEO_ISM_H
00019 #define GNSS_SDR_GALILEO_ISM_H
00020
00021 #include "Galileo_INAV.h"
00022 #include <boost/crc.hpp>
00023 #include <bitset>
00024 #include <cstdint>
00025 #include <unordered_map>
00026 #include <vector>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This class is a storage for the GALILEO Integrity Support Message as described
00036  * in Galileo ICD paragraph 5.2
00037  *
00038  * See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.1.pdf
00039  */
00040 class Galileo_ISM
00041 {
00042 public:
00043     /*!
00044      * Default constructor
00045      */
00046     Galileo_ISM() = default;
00047
00048     void set_ism_constellation_id(uint8_t const_id);
00049     void set_ism_service_level_id(uint8_t sl_id);
00050     void set_ism_wn(uint16_t wn_ism);
00051     void set_ism_t0(uint16_t t0);
00052     void set_ism_mask_msb(bool mask_msb);
00053     void set_ism_mask(uint32_t mask);
00054     void set_ism_pconst(uint8_t pconst);
00055     void set_ism_psat(uint8_t psat);
00056     void set_ism_ura(uint8_t ura);
00057     void set_ism_ure(uint8_t ure);
00058     void set_ism_bnom(uint8_t bnom);
00059     void set_ism_Tvalidity(uint8_t tvalidity);
00060
00061     bool check_ism_crc(const std::bitset<GALILEO_DATA_JK_BITS>& bits);
00062
00063     double get_pconst_value() const;
00064     double get_psat_value() const;
00065     float get_ura_m() const;
00066     float get_ure_m() const;
00067     float get_bnom_m() const;
00068     uint32_t get_mask_ISM() const;
00069     uint16_t get_WN_ISM() const;
00070     uint32_t get_t0_ISM() const;
00071     uint16_t get_Tvalidity_hours() const;
00072     bool get_ism_mask_msb() const;
00073     bool ism_parameters_apply(uint32_t svid) const;
00074
00075 private:
00076     uint32_t compute_crc(const std::vector<uint8_t>& data);
00077     boost::crc_optimal<32, 0x814141AB, 0, 0, false, false> d_crc32_ism;
00078
00079     // ICD 2.1 Table 97
00080     std::unordered_map<uint8_t, double> d_ISM_PCONST_MAP = {
00081         {0, 1.0e-8},
00082         {1, 1.0e-7},
00083         {2, 1.0e-6},
00084         {3, 3.0e-6},
00085         {4, 6.0e-6},
00086         {5, 8.0e-6},
00087         {6, 1.0e-5},
00088         {7, 2.0e-5},
00089         {8, 4.0e-5},
00090         {9, 6.0e-5},
00091         {10, 8.0e-5},
00092         {11, 1.0e-4},
00093         {12, 1.25e-4},
00094         {13, 1.5e-4},
00095         {14, 1.75e-4},
00096         {15, 2.0e-4}};
00097
00098     // ICD 2.1 Table 98
00099     std::unordered_map<uint8_t, double> d_ISM_PSAT_MAP = {
```

```
00100         {0, 1.0e-7},
00101         {1, 3.0e-7},
00102         {2, 6.0e-7},
00103         {3, 1.0e-6},
00104         {4, 2.0e-6},
00105         {5, 3.0e-6},
00106         {6, 5.0e-6},
00107         {7, 7.0e-6},
00108         {8, 1.0e-5},
00109         {9, 1.2e-5},
00110         {10, 1.4e-5},
00111         {11, 1.7e-5},
00112         {12, 2.05e-5},
00113         {13, 2.4e-5},
00114         {14, 2.8e-5},
00115         {15, 3.0e-5}};
00116
00117     // ICD 2.1 Table 99
00118     std::unordered_map<uint8_t, float> d_ISM_URA_MAP = {
00119         {0, 0.75},
00120         {1, 1.0},
00121         {2, 1.5},
00122         {3, 2.0},
00123         {4, 2.25},
00124         {5, 2.50},
00125         {6, 2.75},
00126         {7, 3.0},
00127         {8, 3.25},
00128         {9, 3.50},
00129         {10, 3.75},
00130         {11, 4.0},
00131         {12, 4.50},
00132         {13, 5.0},
00133         {14, 5.50},
00134         {15, 6.0}};
00135
00136     // ICD 2.1 Table 100
00137     std::unordered_map<uint8_t, float> d_ISM_URE_MAP = {
00138         {0, 0.25},
00139         {1, 0.50},
00140         {2, 0.75},
00141         {3, 1.00},
00142         {4, 1.25},
00143         {5, 1.50},
00144         {6, 1.75},
00145         {7, 2.0},
00146         {8, 2.25},
00147         {9, 2.50},
00148         {10, 2.75},
00149         {11, 3.0},
00150         {12, 3.25},
00151         {13, 3.50},
00152         {14, 3.75},
00153         {15, 4.00}};
00154
00155     // ICD 2.1 Table 101
00156     std::unordered_map<uint8_t, float> d_ISM_BNOM_MAP = {
00157         {0, 0.0},
00158         {1, 0.10},
00159         {2, 0.20},
00160         {3, 0.30},
00161         {4, 0.40},
00162         {5, 0.50},
00163         {6, 0.60},
00164         {7, 0.75},
00165         {8, 0.85},
00166         {9, 1.0},
00167         {10, 1.20},
00168         {11, 1.40},
00169         {12, 1.60},
00170         {13, 1.80},
00171         {14, 2.0},
00172         {15, 2.4}};
00173
00174     // ICD 2.1 Table 102
00175     std::unordered_map<uint8_t, uint16_t> d_ISM_TVALIDITY_MAP = {
00176         {0, 1},
00177         {1, 2},
00178         {2, 3},
00179         {3, 4},
00180         {4, 6},
00181         {5, 8},
00182         {6, 12},
00183         {7, 18},
00184         {8, 24},
00185         {9, 36},
00186         {10, 48},
```

```
00187         {11, 72},
00188         {12, 120},
00189         {13, 168},
00190         {14, 720},
00191         {15, 1440}};
00192
00193     uint32_t d_ism_crc{};
00194     uint32_t d_ism_mask{};
00195     uint16_t d_ism_wn{};
00196     uint16_t d_ism_t0{};
00197     uint8_t d_ism_constellation_id{};
00198     uint8_t d_ism_service_level_id{};
00199     uint8_t d_ism_pconst{};
00200     uint8_t d_ism_psat{};
00201     uint8_t d_ism_ura{};
00202     uint8_t d_ism_ure{};
00203     uint8_t d_ism_bnom{};
00204     uint8_t d_ism_Tvalidity{};
00205     bool d_ism_mask_msb{};
00206 };
00207
00208 /** \} */
00209 /** \} */
00210 #endif  // GNSS_SDR_GALILEO_ISM_H
```

13.720 Galileo_OSNMA.h File Reference

Galileo OSNMA message constants.

```
#include <cstddef>
#include <cstdint>
#include <string>
#include <unordered_map>
#include <utility>
#include <vector>
```

Classes

- class Mack_lookup

Functions

- const std::string **PEMFILE_DEFAULT** ("./OSNMA_PublicKey.pem")
- const std::string **CRTFILE_DEFAULT** ("./OSNMA_PublicKey.crt")
- const std::string **MERKLEFILE_DEFAULT** ("./OSNMA_MerkleTree.xml")
- const std::string **KROOTFILE_DEFAULT** ("./OSNMA_DSM_KROOT_NMAHeader.bin")

Variables

- constexpr size_t SIZE_DSM_BLOCKS_BYTES = 13
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_1
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_2
- const std::unordered_map< uint8_t, std::pair< uint16_t, uint16_t > > OSNMA_TABLE_3
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_5
- const std::unordered_map< std::string, uint16_t > OSNMA_TABLE_6
- const std::unordered_map< uint8_t, std::pair< uint16_t, uint16_t > > OSNMA_TABLE_7
- const std::unordered_map< uint8_t, std::string > OSNMA_TABLE_8
- const std::unordered_map< uint8_t, uint16_t > OSNMA_TABLE_10
- const std::unordered_map< uint8_t, uint8_t > OSNMA_TABLE_11
- const std::unordered_map< std::string, uint16_t > OSNMA_TABLE_15
- const std::unordered_map< uint8_t, Mack_lookup > OSNMA_TABLE_16

13.720.1 Detailed Description

Galileo OSNMA message constants.

Author

Carles Fernandez, 2023. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file Galileo_OSNMA.h.

13.721 Galileo_OSNMA.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file Galileo_OSNMA.h
00003  * \brief Galileo OSNMA message constants
00004  * \author Carles Fernandez, 2023. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018 #ifndef GNSS_SDR_GALILEO_OSNMA_H
00019 #define GNSS_SDR_GALILEO_OSNMA_H
00020
00021 #include <cstddef>
00022 #include <cstdint>
00023 #include <string>
00024 #include <unordered_map>
00025 #include <utility>
00026 #include <vector>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033 constexpr size_t SIZE_DSM_BLOCKS_BYTES = 13;
00034
00035 // OSNMA User ICD, Issue 1.1, Table 1
00036 const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_1 = {
00037     {0, std::string("Reserved")},
00038     {1, std::string("Test")},
00039     {2, std::string("Operational")},
00040     {3, std::string("Don't use")}};  // key: nmas, value: nmas status
00041
00042
00043 // OSNMA User ICD, Issue 1.1, Table 2
00044 const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_2 = {
00045     {0, std::string("Reserved")},
00046     {1, std::string("Nominal")},
00047     {2, std::string("End of Chain (EOC)")},
00048     {3, std::string("Chain Revoked (CREV)")},
00049     {4, std::string("New Public Key (NPK)")},
00050     {5, std::string("Public Key Revoked (PKREV)")},
00051     {6, std::string("New Merkle Tree (NMT)")},
00052     {7, std::string("Alert Message (AM)")}};  // key: cpks, value: cpks status
00053
00054 // OSNMA User ICD for the Test Phase, Issue 1.0, Table 3
00055 const std::unordered_map<uint8_t, std::pair<uint16_t, uint16_t>» OSNMA_TABLE_3 = {
00056     {0, {0, 0}},
00057     {1, {0, 0}},
00058     {2, {0, 0}},
00059     {3, {0, 0}},
00060     {4, {0, 0}},
00061     {5, {0, 0}},
00062     {6, {0, 0}},
00063     {7, {13, 1352}},
00064     {8, {14, 1456}},
00065     {9, {15, 1560}},
00066     {10, {16, 1664}},
00067     {11, {0, 0}},
```

```
00068     {12, {0, 0}},
00069     {13, {0, 0}},
00070     {14, {0, 0}},
00071     {15, {0, 0}}};  // key: nb_dp, value: {num_blocks, l_dp_bits}
00072
00073 const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_5 = {
00074     {0, std::string("Reserved")},
00075     {1, std::string("ECDSA P-256")},
00076     {2, std::string("Reserved")},
00077     {3, std::string("ECDSA P-521")},
00078     {4, std::string("OAM")},
00079     {5, std::string("Reserved")},
00080     {6, std::string("Reserved")},
00081     {7, std::string("Reserved")},
00082     {8, std::string("Reserved")},
00083     {9, std::string("Reserved")},
00084     {10, std::string("Reserved")},
00085     {11, std::string("Reserved")},
00086     {12, std::string("Reserved")},
00087     {13, std::string("Reserved")},
00088     {14, std::string("Reserved")},
00089     {15, std::string("Reserved")}};  // key: nptk, value: message
00090
00091 const std::unordered_map<std::string, uint16_t> OSNMA_TABLE_6 = {
00092     {std::string("ECDSA P-256"), 264},
00093     {std::string("ECDSA P-521"), 536}};
00094
00095 // OSNMA User ICD, Issue 1.1, Table 7
00096 const std::unordered_map<uint8_t, std::pair<uint16_t, uint16_t» OSNMA_TABLE_7 = {
00097     {0, {0, 0}},
00098     {1, {7, 728}},
00099     {2, {8, 832}},
00100     {3, {9, 936}},
00101     {4, {10, 1040}},
00102     {5, {11, 1144}},
00103     {6, {12, 1248}},
00104     {7, {13, 1352}},
00105     {8, {14, 1456}},
00106     {9, {0, 0}},
00107     {10, {0, 0}},
00108     {11, {0, 0}},
00109     {12, {0, 0}},
00110     {13, {0, 0}},
00111     {14, {0, 0}},
00112     {15, {0, 0}}};  // key: nb_dk, value: {num_blocks, l_dk_bits}
00113
00114 const std::unordered_map<uint8_t, std::string> OSNMA_TABLE_8 = {
00115     {0, std::string("SHA-256")},
00116     {1, std::string("Reserved")},
00117     {2, std::string("SHA3-256")},
00118     {3, std::string("Reserved")}};  // key: hs, value: hash_function
00119
00120 const std::unordered_map<uint8_t, uint16_t> OSNMA_TABLE_10 = {
00121     {0, 96},
00122     {1, 104},
00123     {2, 112},
00124     {3, 120},
00125     {4, 128},
00126     {5, 160},
00127     {6, 192},
00128     {7, 224},
00129     {8, 256},
00130     {9, 0},
00131     {10, 0},
00132     {11, 0},
00133     {12, 0},
00134     {13, 0},
00135     {14, 0},
00136     {15, 0}};  // key: ks, value: lk_bits
00137
00138 const std::unordered_map<uint8_t, uint8_t> OSNMA_TABLE_11 = {
00139     {0, 0},
00140     {1, 0},
00141     {2, 0},
00142     {3, 0},
00143     {4, 0},
00144     {5, 20},
00145     {6, 24},
00146     {7, 28},
00147     {8, 32},
00148     {9, 40},
00149     {10, 0},
00150     {11, 0},
00151     {12, 0},
00152     {13, 0},
00153     {14, 0},
00154     {15, 0},
```

```
00155 };
00156
00157 const std::unordered_map<std::string, uint16_t> OSNMA_TABLE_15 = {
00158     {std::string("ECDSA P-256"), 512},
00159     {std::string("ECDSA P-521"), 1056}};  // key: ECDSA Curve and hash function, value: {l_ds_bits}
00160
00161 const std::string PEMFILE_DEFAULT("./OSNMA_PublicKey.pem");
00162 const std::string CRTFILE_DEFAULT("./OSNMA_PublicKey.crt");
00163 const std::string MERKLEFILE_DEFAULT("./OSNMA_MerkleTree.xml");
00164 const std::string KROOTFILE_DEFAULT("./OSNMA_DSM_KROOT_NMAHeader.bin");
00165
00166 class Mack_lookup
00167 {
00168 public:
00169     Mack_lookup() = default;
00170     Mack_lookup(uint8_t msg_,
00171         uint8_t nt_,
00172         const std::vector<std::string>& s1_,
00173         const std::vector<std::string>& s2_) : msg(msg_),
00174                                                nt(nt_),
00175                                                sequence1(s1_),
00176                                                sequence2(s2_) {};
00177     uint8_t msg{};
00178     uint8_t nt{};
00179     std::vector<std::string> sequence1;
00180     std::vector<std::string> sequence2;
00181 };
00182
00183 const std::unordered_map<uint8_t, Mack_lookup> OSNMA_TABLE_16 = {
00184     {27, {2, 6, {"00S", "00E", "00E", "00E", "12S", "00E"}, {"00S ", "00E", "00E", "04S", "12S",
      "00E"}}},
00185     {28, {2, 10, {"00S", "00E", "00E", "00E", "00S", "00E", "00E", "12S", "00E", "00E"}, {"00S",
      "00E", "00E", "00S", "00E", "00E", "04S", "12S", "00E", "00E"}}},
00186     {31, {2, 5, {"00S", "00E", "00E", "12S", "00E"}, {"00S", "00E", "00E", "12S", "04S"}}},
00187     {33, {2, 6, {"00S", "00E", "04S", "00E", "12S", "00E"}, {"00S", "00E", "00E", "12S", "00E",
      "12E"}}},
00188     {34, {2, 6, {"00S", "FLX", "04S", "FLX", "12S", "00E"}, {"00S", "FLX", "00E", "12S", "00E",
      "12E"}}},
00189     {35, {2, 6, {"00S", "FLX", "04S", "FLX", "12S", "FLX"}, {"00S", "FLX", "FLX", "12S", "FLX",
      "FLX"}}},
00190     {36, {2, 5, {"00S", "FLX", "04S", "FLX", "12S"}, {"00S", "FLX", "00E", "12S", "12E"}}},
00191     {37, {2, 5, {"00S", "00E", "04S", "00E", "12S"}, {"00S", "00E", "00E", "12S", "12E"}}},
00192     {38, {2, 5, {"00S", "FLX", "04S", "FLX", "12S"}, {"00S", "FLX", "FLX", "12S", "FLX"}}},
00193     {39, {2, 4, {"00S", "FLX", "04S", "FLX"}, {"00S", "FLX", "00E", "12S"}}},
00194     {40, {2, 4, {"00S", "00E", "04S", "12S"}, {"00S", "00E", "00E", "12E"}}},
00195     {41, {2, 4, {"00S", "FLX", "04S", "FLX"}, {"00S", "FLX", "FLX", "12S"}}}};
00196
00197 /** \} */
00198 /** \} */
00199 #endif  // GNSS_SDR_GALILEO_OSNMA_H
```

13.722 galileo_reduced_ced.h File Reference

Galileo Reduced Clock and Ephemeris Data storage class.

```
#include "galileo_ephemeris.h"
#include <cstdint>
```

Classes

- class Galileo_Reduced_CED

  This class manages the Galileo Reduced Clock and Ephemeris Data.

13.722.1 Detailed Description

Galileo Reduced Clock and Ephemeris Data storage class.

Author

Carles Fernandez, 2021. cfernandez(at)cttc.cat

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file galileo_reduced_ced.h.

13.723 galileo_reduced_ced.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_reduced_ced.h
00003  * \brief Galileo Reduced Clock and Ephemeris Data storage class
00004  * \author Carles Fernandez, 2021. cfernandez(at)cttc.cat
00005  *
00006  * -------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GALILEO_REDUCED_CED_H
00019 #define GNSS_SDR_GALILEO_REDUCED_CED_H
00020
00021 #include "galileo_ephemeris.h"
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class manages the Galileo Reduced Clock and Ephemeris Data
00032  */
00033 class Galileo_Reduced_CED
00034 {
00035 public:
00036     /*!
00037      * Default constructor
00038      */
00039     Galileo_Reduced_CED() = default;
00040
00041     /*!
00042      * Convert to Galileo_Ephemeris
00043      */
00044     Galileo_Ephemeris compute_eph() const;
00045
00046     /*!
00047      * \brief Return true when this Reduced CED set is applicable at the given GST epoch.
00048      */
00049     bool is_valid_at(uint32_t week, uint32_t tow) const;
00050
00051     /*!
00052      * \brief Return the Reduced CED orbit and clock reference time of week.
00053      */
00054     uint32_t reference_time() const;
00055
00056     static constexpr uint32_t CED_VALIDITY_SECONDS = 600U;
00057
00058     uint32_t PRN{};         //!< Satellite ID
00059     uint32_t WN{};          //!< GST week containing the start of transmission
00060     uint32_t TOTRedCED{};   //!< Start time of transmission of the Reduced CED word as GST TOW [s]
00061     double DeltaAred{};     //!< Difference between the Reduced CED semi-major axis and the nominal
      semi-major axis [meters]
00062     double exred{};         //!< Reduced CED eccentricity vector component x
00063     double eyred{};         //!< Reduced CED eccentricity vector component y
00064     double Deltai0red{};    //!< Difference between the Reduced CED inclination angle at reference time
      and the nominal inclination [rad]
00065     double Omega0red{};     //!< Reduced CED longitude of ascending node at weekly epoch [rad]
00066     double lambda0red{};    //!< Reduced CED mean argument of latitude [rad]
00067     double af0red{};        //!< Reduced CED satellite clock bias correction coefficient [seconds]
00068     double af1red{};        //!< Reduced CED satellite clock drift correction coefficient
     [seconds/seconds]
00069
00070     // Word 5 data used by the E1/E5b user algorithms. These are not combined
00071     // with full-precision CED orbit or clock parameters.
00072     double BGD_E1E5b{};  //!< E1-E5b Broadcast Group Delay [s]
00073     int32_t E5b_HS{};    //!< E5b Signal Health Status
00074     int32_t E1B_HS{};    //!< E1B Signal Health Status
00075     bool E5b_DVS{};      //!< E5b Data Validity Status
00076     bool E1B_DVS{};      //!< E1B Data Validity Status
00077 };
00078
00079
00080 /** \} */
```

```
00081 /** \} */
00082 #endif  // GNSS_SDR_GALILEO_REDUCED_CED_H
```

13.724 galileo_utc_model.h File Reference

Interface of a Galileo UTC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <cstdint>
```

Classes

- class Galileo_Utc_Model

  This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf paragraph 5.1.7.

13.724.1 Detailed Description

Interface of a Galileo UTC MODEL storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file galileo_utc_model.h.

13.725 galileo_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file galileo_utc_model.h
00003  * \brief  Interface of a Galileo UTC MODEL storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GALILEO_UTC_MODEL_H
00020 #define GNSS_SDR_GALILEO_UTC_MODEL_H
00021
00022 #include <boost/serialization/nvp.hpp>
00023 #include <boost/serialization/version.hpp>
00024 #include <cstdint>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class is a storage for the GALILEO UTC MODEL data as described in Galileo ICD
00034  * https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OS_SIS_ICD_v2.0.pdf
00035  * paragraph 5.1.7
00036  */
00037 class Galileo_Utc_Model
00038 {
```

```
00039 public:
00040     /*!
00041      * Default constructor
00042      */
00043     Galileo_Utc_Model() = default;
00044
00045     // double TOW;
00046     double GST_to_UTC_time(double t_e, int32_t WN) const;  //!< GST-UTC Conversion Algorithm and
      Parameters
00047
00048     /*!
00049      * \brief Signed difference between two truncated week numbers, wrapped to
00050      * [-modulus/2, modulus/2). Implements the week roll-over handling required
00051      * by OS SIS ICD 5.1.7 (modulus 256) and 5.1.8 (modulus 64).
00052      */
00053     static int32_t truncated_week_diff(int32_t wn, int32_t wn_ref, int32_t modulus);
00054
00055     // Word type 6: GST-UTC conversion parameters
00056     double A0{};
00057     double A1{};
00058     int32_t Delta_tLS{};
00059     int32_t tot{};   //!< UTC data reference Time of Week [s]
00060     int32_t WNot{};  //!< UTC data reference Week number [week]
00061     int32_t WN_LSF{};
00062     int32_t DN{};
00063     int32_t Delta_tLSF{};
00064
00065     // GPS to Galileo GST conversion parameters
00066     double A_0G{};
00067     double A_1G{};
00068     int32_t t_0G{};
00069     int32_t WN_0G{};
00070
00071     bool flag_utc_model{};
00072     bool flag_GGTO{};
00073
00074     template <class Archive>
00075
00076     /*!
00077      * \brief Serialize is a boost standard method to be called by the boost XML serialization.
00078      Here is used to save the UTC data on disk file.
00079      */
00080     inline void serialize(Archive& archive, const unsigned int version)
00081     {
00082         archive& BOOST_SERIALIZATION_NVP(A0);
00083         archive& BOOST_SERIALIZATION_NVP(A1);
00084         archive& BOOST_SERIALIZATION_NVP(Delta_tLS);
00085         archive& BOOST_SERIALIZATION_NVP(tot);
00086         archive& BOOST_SERIALIZATION_NVP(WNot);
00087         archive& BOOST_SERIALIZATION_NVP(WN_LSF);
00088         archive& BOOST_SERIALIZATION_NVP(DN);
00089         archive& BOOST_SERIALIZATION_NVP(Delta_tLSF);
00090         archive& BOOST_SERIALIZATION_NVP(flag_utc_model);
00091         if (version > 0)
00092             {
00093                 archive& BOOST_SERIALIZATION_NVP(A_0G);
00094                 archive& BOOST_SERIALIZATION_NVP(A_1G);
00095                 archive& BOOST_SERIALIZATION_NVP(t_0G);
00096                 archive& BOOST_SERIALIZATION_NVP(WN_0G);
00097                 archive& BOOST_SERIALIZATION_NVP(flag_GGTO);
00098             }
00099         else
00100             {
00101                 // Version 0 archives predate GGTO persistence.
00102                 A_0G = 0.0;
00103                 A_1G = 0.0;
00104                 t_0G = 0;
00105                 WN_0G = 0;
00106                 flag_GGTO = false;
00107             }
00108     }
00109 };
00110
00111 BOOST_CLASS_VERSION(Galileo_Utc_Model, 1)
00112
00113
00114 /** \} */
00115 /** \} */
00116 #endif  // GNSS_SDR_GALILEO_UTC_MODEL_H
```

13.726 glonass_gnav_almanac.h File Reference

Interface of a GLONASS GNAV ALMANAC storage.

```
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <array>
#include <cstdint>
```

Classes

- class Glonass_Gnav_Almanac

  This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition 5.1).

13.726.1 Detailed Description

Interface of a GLONASS GNAV ALMANAC storage.

Note

Code added as part of GSoC 2017 program

Author

Damian Miralles, 2017. dmiralles2009(at)gmail.com

See also

GLONASS ICD

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_gnav_almanac.h.

13.727 glonass_gnav_almanac.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_gnav_almanac.h
00003  * \brief  Interface of a GLONASS GNAV ALMANAC storage
00004  * \note Code added as part of GSoC 2017 program
00005  * \author Damian Miralles, 2017. dmiralles2009(at)gmail.com
00006  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_GLONASS_ALMANAC_H
00021 #define GNSS_SDR_GLONASS_ALMANAC_H
00022
00023 #include <boost/serialization/nvp.hpp>
00024 #include <boost/serialization/version.hpp>
00025 #include <array>
00026 #include <cstdint>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This class is a storage for the GLONASS SV ALMANAC data as described GLONASS ICD (Edition
      5.1)
```

```
00036  * \note Code added as part of GSoC 2017 program
00037  * \see <a
       href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
       ICD</a>
00038  */
00039 class Glonass_Gnav_Almanac
00040 {
00041 public:
00042     /*!
00043      * Default constructor
00044      */
00045     Glonass_Gnav_Almanac() = default;
00046
00047     int32_t d_N_A{};  //!< Almanac reference day within its four-year interval (zero if unknown).
00048     int32_t d_N_4{};  //!< Four-year interval containing d_N_A, numbered from 1996.
00049
00050     //! Compute ECEF position in metres, elapsed_s after the reference ascending node (ICD A.3.2).
00051     bool satellite_position(double elapsed_s, std::array<double, 3>& position_m) const;
00052
00053     double d_n_A{};              //!< Conventional number of satellite within GLONASS space segment
       [dimensionless]
00054     double d_H_n_A{};            //!< Carrier frequency number of navigation RF signal transmitted by
       d_nA satellite as table 4.10 (0-31) [dimensionless]
00055     double d_lambda_n_A{};       //!< Longitude of the first (within the d_NA day) ascending node of
       d_nA [radians]
00056     double d_t_lambda_n_A{};     //!< Time of first ascending node passage [s]
00057     double d_Delta_i_n_A{};      //!< Correction of the mean value of inclination of d_n_A satellite
       at instant t_lambda_n_A [radians]
00058     double d_Delta_T_n_A{};      //!< Correction to the mean value of Draconian period of d_n_A
       satellite at instant t_lambda_n_A [s / orbital period]
00059     double d_Delta_T_n_A_dot{};  //!< Rate of change of Draconian period of d_n_A satellite at instant
       t_lambda_n_A [s / orbital period^2]
00060     double d_epsilon_n_A{};      //!< Eccentricity of d_n_A satellite at instant t_lambda_n_A
       [dimensionless]
00061     double d_omega_n_A{};        //!< Argument of perigee of d_n_A satellite at instant t_lambdan_A
       [radians]
00062     double d_M_n_A{};            //!< Type of satellite n_A [dimensionless]
00063     double d_KP{};               //!< Notification on forthcoming leap second correction of UTC
       [dimensionless]
00064     double d_tau_n_A{};          //!< Coarse value of d_n_A satellite time correction to GLONASS time
       at instant  t_lambdan_A[s]
00065     bool d_C_n{};                //!< Generalized “unhealthy flag” of n_A satellite at instant of
       almanac upload [dimensionless]
00066     bool d_l_n{};                //!< Health flag for nth satellite; ln = 0 indicates the n-th
       satellite is helthy, ln = 1 indicates malfunction of this nth satellite [dimensionless]
00067
00068     // Satellite Identification Information
00069     int32_t i_satellite_freq_channel{};  //!< SV Frequency Channel Number
00070     uint32_t PRN{};                      //!< SV PRN Number, equivalent to slot number for
       compatibility with GPS
00071     uint32_t i_satellite_slot_number{};  //!< SV Slot Number
00072
00073     template <class Archive>
00074     /*!
00075      * \brief Serialize is a boost standard method to be called by the boost XML
00076      * serialization. Here is used to save the almanac data on disk file.
00077      */
00078     void serialize(Archive& archive, const uint32_t version)
00079     {
00080         if (version)
00081             {
00082             };
00083
00084         archive& BOOST_SERIALIZATION_NVP(i_satellite_freq_channel);
00085         archive& BOOST_SERIALIZATION_NVP(PRN);
00086         archive& BOOST_SERIALIZATION_NVP(i_satellite_slot_number);
00087         archive& BOOST_SERIALIZATION_NVP(d_n_A);
00088         archive& BOOST_SERIALIZATION_NVP(d_H_n_A);
00089         archive& BOOST_SERIALIZATION_NVP(d_lambda_n_A);
00090         archive& BOOST_SERIALIZATION_NVP(d_t_lambda_n_A);
00091         archive& BOOST_SERIALIZATION_NVP(d_Delta_i_n_A);
00092         archive& BOOST_SERIALIZATION_NVP(d_Delta_T_n_A);
00093         archive& BOOST_SERIALIZATION_NVP(d_Delta_T_n_A_dot);
00094         archive& BOOST_SERIALIZATION_NVP(d_epsilon_n_A);
00095         archive& BOOST_SERIALIZATION_NVP(d_omega_n_A);
00096         archive& BOOST_SERIALIZATION_NVP(d_M_n_A);
00097         archive& BOOST_SERIALIZATION_NVP(d_KP);
00098         archive& BOOST_SERIALIZATION_NVP(d_tau_n_A);
00099         archive& BOOST_SERIALIZATION_NVP(d_C_n);
00100         archive& BOOST_SERIALIZATION_NVP(d_l_n);
00101         if (version > 0)
00102             {
00103                 archive& BOOST_SERIALIZATION_NVP(d_N_A);
00104                 archive& BOOST_SERIALIZATION_NVP(d_N_4);
00105             }
00106     }
00107
```

```
00108 private:
00109     static std::array<double, 6> perturbations(double a, double inclination, double l, double h,
      double longitude, double nt);
00110 };
00111
00112 BOOST_CLASS_VERSION(Glonass_Gnav_Almanac, 1)
00113
00114
00115 /** \} */
00116 /** \} */
00117 #endif  // GNSS_SDR_GLONASS_ALMANAC_H
```

13.728 glonass_gnav_ephemeris.h File Reference

Interface of a GLONASS EPHEMERIS storage.

```
#include "glonass_gnav_utc_model.h"
#include <boost/date_time/posix_time/ptime.hpp>
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

Classes

- class Glonass_Gnav_Ephemeris

  This class is a storage and orbital model functions for the GLONASS SV ephemeris data as described in GLONASS ICD (Edition 5.1).

13.728.1 Detailed Description

Interface of a GLONASS EPHEMERIS storage.

Note

Code added as part of GSoC 2017 program

Author

Damian Miralles, 2017. dmiralles2009(at)gmail.com

See also

GLONASS ICD

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_gnav_ephemeris.h.

13.729 glonass_gnav_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_gnav_ephemeris.h
00003  * \brief  Interface of a GLONASS EPHEMERIS storage
00004  * \note Code added as part of GSoC 2017 program
00005  * \author Damian Miralles, 2017. dmiralles2009(at)gmail.com
00006  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
```

```
00017  */
00018
00019
00020 #ifndef GNSS_SDR_GLONASS_GNAV_EPHEMERIS_H
00021 #define GNSS_SDR_GLONASS_GNAV_EPHEMERIS_H
00022
00023
00024 #include "glonass_gnav_utc_model.h"
00025 #include <boost/date_time/posix_time/ptime.hpp>  // for ptime
00026 #include <boost/serialization/nvp.hpp>
00027 #include <cstdint>
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup System_Parameters
00032  * \{ */
00033
00034
00035 /*!
00036  * \brief This class is a storage and orbital model functions for the GLONASS SV ephemeris data as
      described in GLONASS ICD (Edition 5.1)
00037  * \note Code added as part of GSoC 2017 program
00038  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00039  */
00040 class Glonass_Gnav_Ephemeris
00041 {
00042 public:
00043     /*!
00044      * Default constructor
00045      */
00046     Glonass_Gnav_Ephemeris() = default;
00047
00048     double d_m{};            //!< String number within frame [dimensionless]
00049     double d_t_k{};          //!< GLONASS Time (UTC(SU) + 3 h) referenced to the beginning of the
      frame within the current day [s]
00050     double d_t_b{};          //!< Reference ephemeris relative time in GLONASS Time (UTC(SU) + 3 h).
      Index of a time interval within current day according to UTC(SU) + 03 hours 00 min. [s]
00051     double d_M{};            //!< Type of satellite transmitting navigation signal [dimensionless]
00052     double d_gamma_n{};      //!< Relative deviation of predicted carrier frequency value of n-
      satellite from nominal value at the instant tb [dimensionless]
00053     double d_tau_n{};        //!< Correction to the nth satellite time (tn) relative to GLONASS time
      (te),
00054     double d_Xn{};           //!< Earth-fixed coordinate x of the satellite in PZ-90.02 coordinate
      system [km].
00055     double d_Yn{};           //!< Earth-fixed coordinate y of the satellite in PZ-90.02 coordinate
      system [km]
00056     double d_Zn{};           //!< Earth-fixed coordinate z of the satellite in PZ-90.02 coordinate
      system [km]
00057     double d_VXn{};          //!< Earth-fixed velocity coordinate x of the satellite in PZ-90.02
      coordinate system [km/s]
00058     double d_VYn{};          //!< Earth-fixed velocity coordinate y of the satellite in PZ-90.02
      coordinate system [km/s]
00059     double d_VZn{};          //!< Earth-fixed velocity coordinate z of the satellite in PZ-90.02
      coordinate system [km/s]
00060     double d_AXn{};          //!< Earth-fixed acceleration coordinate x of the satellite in PZ-90.02
      coordinate system [km/s^2]
00061     double d_AYn{};          //!< Earth-fixed acceleration coordinate y of the satellite in PZ-90.02
      coordinate system [km/s^2]
00062     double d_AZn{};          //!< Earth-fixed acceleration coordinate z of the satellite in PZ-90.02
      coordinate system [km/s^2]
00063     double d_B_n{};          //!< Health flag [dimensionless]
00064     double d_P{};            //!< Technological parameter of control segment, indication the satellite
      operation mode in respect of time parameters [dimensionless]
00065     double d_N_T{};          //!< Current date, calendar number of day within four-year interval
      starting from the 1-st of January in a leap year [days]
00066     double d_F_T{};          //!< Parameter that provides the predicted satellite user range accuracy
      at time tb [dimensionless]
00067     double d_n{};            //!< Index of the satellite transmitting given navigation signal. It
      corresponds to a slot number within GLONASS constellation
00068     double d_Delta_tau_n{};  //!< Time difference between navigation RF signal transmitted in L2 sub-
      band and aviation RF signal transmitted in L1 sub-band by nth satellite. [dimensionless]
00069     double d_E_n{};          //!< Characterises "age" of a current information [days]
00070     double d_P_1{};          //!< Flag of the immediate data updating [minutes]
00071     bool d_P_2{};            //!< Flag of oddness ("1") or evenness ("0") of the value of (tb)
      [dimensionless]
00072     bool d_P_3{};            //!< Flag indicating a number of satellites for which almanac is
      transmitted within given frame: "1" corresponds to 5 satellites and "0" corresponds to 4 satellites
      [dimensionless]
00073     bool d_P_4{};            //!< Flag to show that ephemeris parameters are present. "1" indicates
      that updated ephemeris or frequency/time parameters have been uploaded by the control segment
      [dimensionless]
00074     bool d_l3rd_n{};         //!< Health flag for nth satellite; ln = 0 indicates the n-th satellite
      is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless]
00075     bool d_l5th_n{};         //!< Health flag for nth satellite; ln = 0 indicates the n-th satellite
      is healthy, ln = 1 indicates malfunction of this nth satellite [dimensionless]
```

```
00076
00077     // Immediate deliverables of ephemeris information
00078     // Satellite Identification Information
00079     int32_t i_satellite_freq_channel{};  //!< SV Frequency Channel Number
00080     uint32_t PRN{};                      //!< SV PRN Number, equivalent to slot number for
      compatibility with GPS
00081     uint32_t i_satellite_slot_number{};  //!< SV Slot Number
00082     double d_yr = 1972.0;                //!< Current year
00083     double d_satClkDrift{};              //!< GLONASS clock error
00084     double d_dtr{};                      //!< relativistic clock correction term
00085     double d_iode{};                     //!< Issue of data, ephemeris (Bit 0-6 of tb)
00086     double d_tau_c{};                    //!< GLONASST 2 UTC correction (todo) may be eliminated
00087     double d_TOW{};                      //!< GLONASST IN GPST seconds of week
00088     int32_t d_WN{};                      //!< GLONASST IN GPST week number of the start of frame
00089     double d_tod{};                      //!< Time of Day since ephemeris where decoded
00090     double d_kp_leap_correction_s{};     //!< UTC leap second announced by the KP word and already in
      effect, but not yet in the receiver's leap second table [s]
00091
00092     /*!
00093      * \brief Sets (\a d_satClkDrift)and returns the clock drift in seconds according to the User
      Algorithm for SV Clock Correction
00094      */
00095     double sv_clock_drift(double transmitTime, double timeCorrUTC);
00096
00097     /*!
00098      *  \brief Computes the GLONASS System Time and returns a boost::posix_time::ptime object
00099      * \ param offset_time Is the start of day offset to compute the time
00100      */
00101     boost::posix_time::ptime compute_GLONASS_time(double offset_time) const;
00102
00103     /*!
00104      * \brief Converts from GLONASST to UTC
00105      * \details The function simply adjust for the 6 hrs offset between GLONASST and UTC
00106      * \param[in] offset_time Is the start of day offset
00107      * \param[in] glot2utc_corr Correction from GLONASST to UTC
00108      * \returns UTC time as a boost::posix_time::ptime object
00109      */
00110     boost::posix_time::ptime glot_to_utc(double offset_time, double glot2utc_corr) const;
00111
00112     /*!
00113      * \brief Converts from GLONASST to GPST
00114      * \details Converts from GLONASST to GPST in time of week (TOW) and week number (WN) format
00115      * \param[in] tod_offset Is the start of day offset
00116      * \param[in] glot2utc_corr Correction from GLONASST to UTC
00117      * \param[in] glot2gpst_corr Correction from GLONASST to GPST
00118      * \param[out] WN Week Number, not in mod(1024) format
00119      * \param[out] TOW Time of Week in seconds of week
00120      */
00121     void glot_to_gpst(double tod_offset, double glot2utc_corr, double glot2gpst_corr, int32_t* WN,
      double* TOW) const;
00122
00123     template <class Archive>
00124
00125     /*!
00126      * \brief Serialize is a boost standard method to be called by the boost XML
00127      * serialization. Here is used to save the ephemeris data on disk file.
00128      */
00129     void serialize(Archive& archive, const uint32_t version)
00130     {
00131         if (version)
00132             {
00133             };
00134
00135         archive& BOOST_SERIALIZATION_NVP(i_satellite_freq_channel);  //!< SV PRN frequency channel
      number
00136         archive& BOOST_SERIALIZATION_NVP(PRN);
00137         archive& BOOST_SERIALIZATION_NVP(i_satellite_slot_number);
00138         archive& BOOST_SERIALIZATION_NVP(d_m);                    //!< String number within frame
      [dimensionless]
00139         archive& BOOST_SERIALIZATION_NVP(d_t_k);                  //!< Time referenced to the
      beginning of the frame within the current day [hours, minutes, seconds]
00140         archive& BOOST_SERIALIZATION_NVP(d_t_b);                  //!< Index of a time interval
      within current day according to UTC(SU) + 03 hours 00 min. [minutes]
00141         archive& BOOST_SERIALIZATION_NVP(d_M);                    //!< Type of satellite transmitting
      navigation signal [dimensionless]
00142         archive& BOOST_SERIALIZATION_NVP(d_gamma_n);              //!< Relative deviation of
      predicted carrier frequency value of n- satellite from nominal value at the instant tb [dimensionless]
00143         archive& BOOST_SERIALIZATION_NVP(d_tau_n);                //!< Correction to the nth
      satellite time (tn) relative to GLONASS time (te)
00144         archive& BOOST_SERIALIZATION_NVP(d_Xn);                   //!< Earth-fixed coordinate x of
      the satellite in PZ-90.02 coordinate system [km].
00145         archive& BOOST_SERIALIZATION_NVP(d_Yn);                   //!< Earth-fixed coordinate y of
      the satellite in PZ-90.02 coordinate system [km]
00146         archive& BOOST_SERIALIZATION_NVP(d_Zn);                   //!< Earth-fixed coordinate z of
      the satellite in PZ-90.02 coordinate system [km]
00147         archive& BOOST_SERIALIZATION_NVP(d_VXn);                  //!< Earth-fixed velocity
      coordinate x of the satellite in PZ-90.02 coordinate system [km/s]
```

```
00148         archive& BOOST_SERIALIZATION_NVP(d_VYn);                    //!< Earth-fixed velocity
      coordinate y of the satellite in PZ-90.02 coordinate system [km/s]
00149         archive& BOOST_SERIALIZATION_NVP(d_VZn);                    //!< Earth-fixed velocity
      coordinate z of the satellite in PZ-90.02 coordinate system [km/s]
00150         archive& BOOST_SERIALIZATION_NVP(d_AXn);                    //!< Earth-fixed acceleration
      coordinate x of the satellite in PZ-90.02 coordinate system [km/s^2]
00151         archive& BOOST_SERIALIZATION_NVP(d_AYn);                    //!< Earth-fixed acceleration
      coordinate y of the satellite in PZ-90.02 coordinate system [km/s^2]
00152         archive& BOOST_SERIALIZATION_NVP(d_AZn);                    //!< Earth-fixed acceleration
      coordinate z of the satellite in PZ-90.02 coordinate system [km/s^2]
00153         archive& BOOST_SERIALIZATION_NVP(d_B_n);                    //!< Health flag [dimensionless]
00154         archive& BOOST_SERIALIZATION_NVP(d_P);                      //!< Technological parameter of
      control segment, indication the satellite operation mode in respect of time parameters [dimensionless]
00155         archive& BOOST_SERIALIZATION_NVP(d_N_T);                    //!< Current date, calendar number
      of day within four-year interval starting from the 1-st of January in a leap year [days]
00156         archive& BOOST_SERIALIZATION_NVP(d_F_T);                    //!< Parameter that provides the
      predicted satellite user range accuracy at time tb [dimensionless]
00157         archive& BOOST_SERIALIZATION_NVP(d_n);                      //!< Index of the satellite
      transmitting given navigation signal. It corresponds to a slot number within GLONASS constellation
00158         archive& BOOST_SERIALIZATION_NVP(d_Delta_tau_n);            //!< Time difference between
      navigation RF signal transmitted in L2 sub- band and aviation RF signal transmitted in L1 sub-band by
      nth satellite. [dimensionless]
00159         archive& BOOST_SERIALIZATION_NVP(d_E_n);                    //!< Characterises "age" of a
      current information [days]
00160         archive& BOOST_SERIALIZATION_NVP(d_P_1);                    //!< Flag of the immediate data
      updating.
00161         archive& BOOST_SERIALIZATION_NVP(d_P_2);                    //!< Flag of oddness ("1") or
      evenness ("0") of the value of (tb) [dimensionless]
00162         archive& BOOST_SERIALIZATION_NVP(d_P_3);                    //!< Flag indicating a number of
      satellites for which almanac is transmitted within given frame: "1" corresponds to 5 satellites and
      "0" corresponds to 4 satellites [dimensionless]
00163         archive& BOOST_SERIALIZATION_NVP(d_P_4);                    //!< Flag to show that ephemeris
      parameters are present. "1" indicates that updated ephemeris or frequency/time parameters have been
      uploaded by the control segment [dimensionless]
00164         archive& BOOST_SERIALIZATION_NVP(d_l3rd_n);                 //!< Health flag for nth satellite;
      ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite
      [dimensionless]
00165         archive& BOOST_SERIALIZATION_NVP(d_l5th_n);                 //!< Health flag for nth satellite;
      ln = 0 indicates the n-th satellite is helthy, ln = 1 indicates malfunction of this nth satellite
      [dimensionless]
00166         archive& BOOST_SERIALIZATION_NVP(d_kp_leap_correction_s);  //!< UTC leap second announced by
      the KP word and already in effect [s]
00167     }
00168
00169 private:
00170     /*
00171      * Accounts for the beginning or end of week crossover
00172      *
00173      * \param[in]  -  time in seconds
00174      * \param[out] -  corrected time, in seconds
00175      */
00176     double check_t(double time);
00177 };
00178
00179
00180 /** \} */
00181 /** \} */
00182 #endif  // GNSS_SDR_GLONASS_GNAV_EPHEMERIS_H
```

13.730 glonass_gnav_navigation_message.h File Reference

Interface of a GLONASS GNAV Data message decoder as described in GLONASS ICD (Edition 5.1).

```
#include "GLONASS_L1_L2_CA.h"
#include "glonass_gnav_almanac.h"
#include "glonass_gnav_ephemeris.h"
#include "glonass_gnav_utc_model.h"
#include <boost/date_time/posix_time/ptime.hpp>
#include <bitset>
#include <cstdint>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Glonass_Gnav_Navigation_Message

This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1).

13.730.1 Detailed Description

Interface of a GLONASS GNAV Data message decoder as described in GLONASS ICD (Edition 5.1).

Note

Code added as part of GSoC 2017 program

Author

Damian Miralles, 2017. dmiralles2009(at)gmail.com

See also

GLONASS ICD

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_gnav_navigation_message.h.

13.731 glonass_gnav_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_gnav_navigation_message.h
00003  * \brief  Interface of a GLONASS GNAV Data message decoder as described in GLONASS ICD (Edition 5.1)
00004  * \note Code added as part of GSoC 2017 program
00005  * \author Damian Miralles, 2017. dmiralles2009(at)gmail.com
00006  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_GLONASS_GNAV_NAVIGATION_MESSAGE_H
00021 #define GNSS_SDR_GLONASS_GNAV_NAVIGATION_MESSAGE_H
00022
00023
00024 #include "GLONASS_L1_L2_CA.h"
00025 #include "glonass_gnav_almanac.h"
00026 #include "glonass_gnav_ephemeris.h"
00027 #include "glonass_gnav_utc_model.h"
00028 #include <boost/date_time/posix_time/ptime.hpp>
00029 #include <bitset>
00030 #include <cstdint>
00031 #include <string>
00032 #include <utility>  // for pair
00033 #include <vector>   // for vector
00034
00035 /** \addtogroup Core
00036  * \{ */
00037 /** \addtogroup System_Parameters
00038  * \{ */
00039
00040
00041 /*!
00042  * \brief This class decodes a GLONASS GNAV Data message as described in GLONASS ICD (Edition 5.1)
00043  * \note Code added as part of GSoC 2017 program
00044  * \see <a
      href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
      ICD</a>
00045  */
00046 class Glonass_Gnav_Navigation_Message
00047 {
00048 public:
```

```
00049     /*!
00050      * Default constructor
00051      */
00052     Glonass_Gnav_Navigation_Message() = default;
00053
00054     /*!
00055      * \brief Compute CRC for GLONASS GNAV strings
00056      * \param bits Bits of the string message where to compute CRC
00057      */
00058     bool CRC_test(std::bitset<GLONASS_GNAV_STRING_BITS>& bits) const;
00059
00060     /*!
00061      * \brief Computes the frame number being decoded given the satellite slot number
00062      * \param satellite_slot_number [in] Satellite slot number identifier
00063      * \returns Frame number being decoded, 0 if operation was not successful.
00064      */
00065     uint32_t get_frame_number(uint32_t satellite_slot_number);
00066
00067     /*!
00068      * \brief Obtain a GLONASS GNAV SV Ephemeris class filled with current SV data
00069      */
00070     Glonass_Gnav_Ephemeris get_ephemeris() const
00071     {
00072         return gnav_ephemeris;
00073     }
00074
00075     /*!
00076      * \brief Obtain a GLONASS GNAV UTC model parameters class filled with current SV data
00077      */
00078     inline Glonass_Gnav_Utc_Model get_utc_model() const
00079     {
00080         return gnav_utc_model;
00081     }
00082
00083     /*!
00084      * \brief Returns a Glonass_Gnav_Almanac object filled with the latest navigation data received
00085      * \param satellite_slot_number Slot number identifier for the satellite
00086      * \returns Returns the Glonass_Gnav_Almanac object for the input slot number
00087      */
00088     Glonass_Gnav_Almanac get_almanac(uint32_t satellite_slot_number) const;
00089
00090     /*!
00091      * \brief Returns true if a new Glonass_Gnav_Ephemeris object has arrived.
00092      */
00093     bool have_new_ephemeris();
00094
00095     /*!
00096      * \brief Returns true if new Glonass_Gnav_Utc_Model object has arrived
00097      */
00098     bool have_new_utc_model();
00099
00100     /*!
00101      * \brief Returns true if new Glonass_Gnav_Almanac object has arrived.
00102      */
00103     bool have_new_almanac();
00104
00105     /*!
00106      * \brief Decodes the GLONASS GNAV string
00107      * \param frame_string [in] is the string message within the parsed frame
00108      * \returns Returns the ID of the decoded string
00109      */
00110     int32_t string_decoder(const std::string& frame_string);
00111
00112     inline bool get_flag_CRC_test() const
00113     {
00114         return flag_CRC_test;
00115     }
00116
00117     inline void set_rf_link(int32_t rf_link)
00118     {
00119         gnav_ephemeris.i_satellite_freq_channel = rf_link;
00120     }
00121
00122     inline uint32_t get_alm_satellite_slot_number() const
00123     {
00124         return i_alm_satellite_slot_number;
00125     }
00126
00127     inline bool get_flag_update_slot_number() const
00128     {
00129         return flag_update_slot_number;
00130     }
00131
00132     inline void set_flag_update_slot_number(bool flag_slot)
00133     {
00134         flag_update_slot_number = flag_slot;
00135     }
```

```
00136
00137     inline bool get_flag_TOW_new() const
00138     {
00139         return flag_TOW_new;
00140     }
00141
00142     inline void set_flag_TOW_new(bool tow_new)
00143     {
00144         flag_TOW_new = tow_new;
00145     }
00146
00147     inline bool is_flag_TOW_set() const
00148     {
00149         return flag_TOW_set;
00150     }
00151
00152     inline void set_flag_TOW_set(bool tow_set)
00153     {
00154         flag_TOW_set = tow_set;
00155     }
00156
00157     inline void set_flag_ephemeris_str_1(bool ephemeris_str_1)
00158     {
00159         flag_ephemeris_str_1 = ephemeris_str_1;
00160     }
00161
00162     inline void set_flag_ephemeris_str_2(bool ephemeris_str_2)
00163     {
00164         flag_ephemeris_str_2 = ephemeris_str_2;
00165     }
00166
00167     inline void set_flag_ephemeris_str_3(bool ephemeris_str_3)
00168     {
00169         flag_ephemeris_str_3 = ephemeris_str_3;
00170     }
00171
00172     inline void set_flag_ephemeris_str_4(bool ephemeris_str_4)
00173     {
00174         flag_ephemeris_str_4 = ephemeris_str_4;
00175     }
00176
00177 private:
00178     uint64_t read_navigation_unsigned(const std::bitset<GLONASS_GNAV_STRING_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00179     int64_t read_navigation_signed(const std::bitset<GLONASS_GNAV_STRING_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00180     bool read_navigation_bool(const std::bitset<GLONASS_GNAV_STRING_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00181     void update_almanac_satellite_info();
00182     void update_kp_leap_second_state();
00183     void check_kp_leap_event();
00184     void apply_kp_leap_second(int32_t leap_s);
00185
00186     boost::posix_time::ptime d_kp_leap_event_utc;  // UTC epoch of the leap second event announced by
      KP, while armed
00187
00188     Glonass_Gnav_Ephemeris gnav_ephemeris{};                    // Ephemeris information decoded
00189     Glonass_Gnav_Utc_Model gnav_utc_model{};                    // UTC model information
00190     Glonass_Gnav_Almanac gnav_almanac[GLONASS_CA_NBR_SATS]{};  // Almanac information for all 24
      satellites
00191
00192     double d_previous_tb{};                        // Previous iode for the Glonass_Gnav_Ephemeris
      object. Used to determine when new data arrives
00193     double d_previous_Na[GLONASS_CA_NBR_SATS]{};  // Previous time for almanac of the
      Glonass_Gnav_Almanac object
00194
00195     uint32_t d_frame_ID{};
00196     uint32_t d_string_ID{};
00197     uint32_t i_alm_satellite_slot_number{};  // SV Orbit Slot Number
00198     // Satellite slot number validated by an even almanac string (6/8/10/12/14)
00199     // and consumed by its paired odd string (7/9/11/13/15). Kept per pair so a
00200     // stale flag can never complete an almanac with a slot from another pair.
00201     uint32_t d_alm_slot_str_6{};
00202     uint32_t d_alm_slot_str_8{};
00203     uint32_t d_alm_slot_str_10{};
00204     uint32_t d_alm_slot_str_12{};
00205     uint32_t d_alm_slot_str_14{};
00206
00207     int32_t d_kp_announced_leap_s{};  // UTC correction announced by KP (+1 or -1 s), 0 if not armed
00208
00209     bool flag_CRC_test{};
00210     bool flag_update_slot_number{};
00211
00212     // Ephemeris Flags and control variables
00213     bool flag_all_ephemeris{};    // Flag indicating that all strings containing ephemeris have been
      received
00214     bool flag_ephemeris_str_1{};  // Flag indicating that ephemeris 1/4 (string 1) have been received
```

```
00215     bool flag_ephemeris_str_2{};  // Flag indicating that ephemeris 2/4 (string 2) have been received
00216     bool flag_ephemeris_str_3{};  // Flag indicating that ephemeris 3/4 (string 3) have been received
00217     bool flag_ephemeris_str_4{};  // Flag indicating that ephemeris 4/4 (string 4) have been received
00218
00219     // Almanac Flags
00220     bool flag_almanac_str_6{};   // Flag indicating that almanac of string 6 have been received
00221     bool flag_almanac_str_7{};   // Flag indicating that almanac of string 7 have been received
00222     bool flag_almanac_str_8{};   // Flag indicating that almanac of string 8 have been received
00223     bool flag_almanac_str_9{};   // Flag indicating that almanac of string 9 have been received
00224     bool flag_almanac_str_10{};  // Flag indicating that almanac of string 10 have been received
00225     bool flag_almanac_str_11{};  // Flag indicating that almanac of string 11 have been received
00226     bool flag_almanac_str_12{};  // Flag indicating that almanac of string 12 have been received
00227     bool flag_almanac_str_13{};  // Flag indicating that almanac of string 13 have been received
00228     bool flag_almanac_str_14{};  // Flag indicating that almanac of string 14 have been received
00229     bool flag_almanac_str_15{};  // Flag indicating that almanac of string 15 have been received
00230
00231     // UTC and System Clocks Flags
00232     bool flag_utc_model_str_5{};  // Clock info send in string 5 of navigation data
00233
00234     bool flag_TOW_set{};  // Flag indicating when the TOW has been set
00235     bool flag_TOW_new{};  // Flag indicating when a new TOW has been computed
00236 };
00237
00238
00239 /** \} */
00240 /** \} */
00241 #endif  // GNSS_SDR_GLONASS_GNAV_NAVIGATION_MESSAGE_H
```

13.732 glonass_gnav_utc_model.h File Reference

Interface of a GLONASS GNAV UTC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

Classes

- class Glonass_Gnav_Utc_Model

  This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD (Edition 5.1).

13.732.1 Detailed Description

Interface of a GLONASS GNAV UTC MODEL storage.

Note

Code added as part of GSoC 2017 program

Author

Damian Miralles, 2017. dmiralles2009(at)gmail.com

See also

GLONASS ICD

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file glonass_gnav_utc_model.h.

13.733 glonass_gnav_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file glonass_gnav_utc_model.h
00003  * \brief  Interface of a GLONASS GNAV UTC MODEL storage
00004  * \note Code added as part of GSoC 2017 program
00005  * \author Damian Miralles, 2017. dmiralles2009(at)gmail.com
```

```
00006  * \see <a
       href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
       ICD</a>
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019
00020 #ifndef GNSS_SDR_GLONASS_GNAV_UTC_MODEL_H
00021 #define GNSS_SDR_GLONASS_GNAV_UTC_MODEL_H
00022
00023 #include <boost/serialization/nvp.hpp>
00024 #include <cstdint>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class is a storage for the GLONASS GNAV UTC MODEL data as described in GLONASS ICD
       (Edition 5.1)
00034  * \note Code added as part of GSoC 2017 program
00035  * \see <a
       href="http://russianspacesystems.ru/wp-content/uploads/2016/08/ICD_GLONASS_eng_v5.1.pdf">GLONASS
       ICD</a>
00036  */
00037 class Glonass_Gnav_Utc_Model
00038 {
00039 public:
00040     /*!
00041      * Default constructor
00042      */
00043     Glonass_Gnav_Utc_Model() = default;
00044
00045     bool valid{};
00046     // Clock Parameters
00047     double d_tau_c{};     //!< GLONASS time scale correction to UTC(SU) time. [s]
00048     double d_tau_gps{};   //!< Correction to GPS time to GLONASS time [day]
00049     double d_N_4{};       //!< Four year interval number starting from 1996 [4 year interval]
00050     double d_N_A{};       //!< Calendar day number within the four-year period beginning since the leap
       year for Almanac data [days]
00051     double d_B1{};        //!< Coefficient  to  determine DeltaUT1 [s]
00052     double d_B2{};        //!< Coefficient  to  determine DeltaUT1 [s/msd]
00053     double d_KP{};        //!< Notification of forthcoming leap second correction of UTC (GLONASS ICD
       Table 4.12) [dimensionless]
00054
00055     /*!
00056      * \brief Computes the Coordinated Universal Time (UTC) and
00057      * returns it in [s] (GLONASS ICD (Edition 5.1) Section 3.3.3 GLONASS Time)
00058      */
00059     double utc_time(double glonass_time_corrected) const;
00060
00061     /*!
00062      * \brief Returns the UTC leap second correction announced by the KP word
00063      * for the end of the current quarter: +1 s, -1 s, or 0 if no correction
00064      * is planned (GLONASS ICD Edition 5.1, Table 4.12)
00065      */
00066     inline int32_t announced_leap_second() const
00067     {
00068         if (static_cast<int32_t>(d_KP) == 1)
00069             {
00070                 return 1;
00071             }
00072         if (static_cast<int32_t>(d_KP) == 3)
00073             {
00074                 return -1;
00075             }
00076         return 0;
00077     }
00078
00079     template <class Archive>
00080     /*!
00081      * \brief Serialize is a boost standard method to be called by the boost XML
00082      * serialization. Here is used to save the almanac data on disk file.
00083      */
00084     void serialize(Archive& archive, const uint32_t version)
00085     {
```

```
00086         if (version)
00087             {
00088             };
00089         archive& BOOST_SERIALIZATION_NVP(valid);
00090         archive& BOOST_SERIALIZATION_NVP(d_tau_c);
00091         archive& BOOST_SERIALIZATION_NVP(d_tau_gps);
00092         archive& BOOST_SERIALIZATION_NVP(d_N_4);
00093         archive& BOOST_SERIALIZATION_NVP(d_N_A);
00094         archive& BOOST_SERIALIZATION_NVP(d_B1);
00095         archive& BOOST_SERIALIZATION_NVP(d_B2);
00096         archive& BOOST_SERIALIZATION_NVP(d_KP);
00097     }
00098 };
00099
00100
00101 /** \} */
00102 /** \} */
00103 #endif  // GNSS_SDR_GLONASS_GNAV_UTC_MODEL_H
```

13.734 GLONASS_L1_L2_CA.h File Reference

Defines system parameters for GLONASS L1 C/A signal and NAV data.

```
#include "gnss_frequencies.h"
#include <cstdint>
#include <map>
#include <utility>
#include <vector>
```

Macros

- #define GLONASS_GNAV_PREAMBLE
- #define GLONASS_GNAV_PREAMBLE_SAMPLES

Functions

- const std::vector< std::pair< int32_t, int32_t > > **STRING_ID** ({{2, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **KX** ({{78, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **P1** ({{8, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_HR** ({{10, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_MIN** ({{15, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **T_K_SEC** ({{21, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **X_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **B_N** ({{6, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **P2** ({{9, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **T_B** ({{10, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **Y_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **P3** ({{6, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **GAMMA_N** ({{7, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **P** ({{19, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **EPH_L_N** ({{21, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N_DOT** ({{22, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N_DOT_DOT** ({{46, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **Z_N** ({{51, 27}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_N** ({{6, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_TAU_N** ({{28, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **E_N** ({{33, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **P4** ({{52, 1}})

- const std::vector< std::pair< int32_t, int32_t > > **F_T** ({{53, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **N_T** ({{60, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **N** ({{71, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **M** ({{76, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **DAY_NUMBER_A** ({{6, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_C** ({{17, 32}})
- const std::vector< std::pair< int32_t, int32_t > > **N_4** ({{50, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_GPS** ({{55, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_L_N** ({{77, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **C_N** ({{6, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **M_N_A** ({{7, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **N_A** ({{9, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **TAU_N_A** ({{14, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **LAMBDA_N_A** ({{24, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_I_N_A** ({{45, 18}})
- const std::vector< std::pair< int32_t, int32_t > > **EPSILON_N_A** ({{63, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_N_A** ({{6, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **T_LAMBDA_N_A** ({{22, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_N_A** ({{43, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_T_DOT_N_A** ({{65, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **H_N_A** ({{72, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **B1** ({{6, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **B2** ({{17, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **KP** ({{27, 2}})

Variables

- constexpr double GLONASS_F_M_A = 0.35e9

  Gravitational constant of atmosphere [m^3/s^2].
- constexpr double GLONASS_SEMI_MAJOR_AXIS = 6378136

  Semi-major axis of Earth [m].
- constexpr double GLONASS_FLATTENING = 1.0 / 29825784.0

  Flattening parameter.
- constexpr double GLONASS_GRAVITY = 97803284.0

  Equatorial acceleration of gravity [mGal].
- constexpr double GLONASS_GRAVITY_CORRECTION = 0.87

  Correction to acceleration of gravity at sea-level due to Atmosphere[uGal].
- constexpr double GLONASS_J2 = 1082625.75e-9

  Second zonal harmonic of the geopotential.
- constexpr double GLONASS_J4 = -2370.89e-9

  Fourth zonal harmonic of the geopotential.
- constexpr double GLONASS_J6 = 6.08e-9

  Sixth zonal harmonic of the geopotential.
- constexpr double GLONASS_J8 = 1.40e-11

  Eighth zonal harmonic of the geopotential.
- constexpr double GLONASS_U0 = 62636861.4

  Normal potential at surface of common terrestrial ellipsoid [m^2/s^2].
- constexpr double GLONASS_C20 = -1082.63e-6

  Second zonal coefficient of spherical harmonic expansion.
- constexpr double GLONASS_EARTH_RADIUS = 6378.136

  Equatorial radius of Earth [km].
- constexpr double GLONASS_EARTH_INCLINATION = 0.000409148809899e3

  Mean inclination of ecliptic to equator (23 deg 26 min 33 sec) [rad].

- constexpr double GLONASS_TAU_0 = -0.005835151531174e3

  (-334 deg 19 min 46.40 sec) [rad];
- constexpr double GLONASS_TAU_1 = 0.071018041257371e3

  (4069 deg 02 min 02.52 sec) [rad];
- constexpr double GLONASS_MOON_Q0 = -0.001115184961435e3

  (-63 deg 53 min 43.41 sec) [rad]
- constexpr double GLONASS_MOON_Q1 = 8.328691103668023e3

  (477198 deg 50 min 56.79 sec) [rad]
- constexpr double GLONASS_MOON_OMEGA_0 = 0.004523601514852e3

  (259 deg 10 min 59.79 sec) [rad]
- constexpr double GLONASS_MOON_OMEGA_1 = -0.033757146246552e3

  (-1934 deg 08 min 31.23 sec) [rad]
- constexpr double GLONASS_MOON_GM = 4902.835

  Lunar gravitational constant [km^3/s^2].
- constexpr double GLONASS_MOON_SEMI_MAJOR_AXIS = 3.84385243e5

  Semi-major axis of lunar orbit [km];.
- constexpr double GLONASS_MOON_ECCENTRICITY = 0.054900489

  Eccentricity of lunar orbit.
- constexpr double GLONASS_MOON_INCLINATION = 0.000089803977407e3

  Inclination of lunar orbit to ecliptic plane (5 deg 08 min 43.4 sec) [rad].
- constexpr double GLONASS_SUN_OMEGA = 0.004908229466869e3

  TODO What is this operation in the seconds with T?(281 deg 13 min 15.0 + 6189.03 x T sec) [rad].
- constexpr double GLONASS_SUN_Q0 = 0.006256583774423e3

  (358 deg 28 min 33.04 sec) [rad]
- constexpr double GLONASS_SUN_Q1 = 0e3

  TODO Why is the value greater than 60?(129596579.10 sec) [rad].
- constexpr double GLONASS_SUN_GM = 0.1325263e12

  Solar gravitational constant [km^3/s^2].
- constexpr double GLONASS_SUN_SEMI_MAJOR_AXIS = 1.49598e8

  Semi-major axis of solar orbit [km];.
- constexpr double GLONASS_SUN_ECCENTRICITY = 0.016719

  Eccentricity of solar orbit.
- constexpr double GLONASS_L2_CA_FREQ_HZ = FREQ2_GLO

  L2 [Hz].
- constexpr double GLONASS_L2_CA_DFREQ_HZ = DFRQ2_GLO

  Freq Bias for GLONASS L1 [Hz].
- constexpr double GLONASS_L2_CA_CODE_RATE_CPS = 0.511e6

  GLONASS L1 C/A code rate [chips/s].
- constexpr double GLONASS_L2_CA_CODE_LENGTH_CHIPS = 511.0

  GLONASS L1 C/A code length [chips].
- constexpr double GLONASS_L2_CA_CODE_PERIOD_S = 0.001

  GLONASS L1 C/A code period [seconds].
- constexpr double GLONASS_L2_CA_SYMBOL_PERIOD_S = 0.01

  GLONASS L1 C/A symbol period [seconds].
- constexpr double GLONASS_L2_CA_CHIP_PERIOD_S = 1.9569e-06

  GLONASS L1 C/A chip period [seconds].
- constexpr double GLONASS_L2_CA_SYMBOL_RATE_BPS = 1000.0
- constexpr double GLONASS_L1_CA_FREQ_HZ = FREQ1_GLO

  L1 [Hz].
- constexpr double GLONASS_L1_CA_DFREQ_HZ = DFRQ1_GLO

  Freq Bias for GLONASS L1 [Hz].

- constexpr double GLONASS_L1_CA_CODE_RATE_CPS = 0.511e6

  GLONASS L1 C/A code rate [chips/s].
- constexpr double GLONASS_L1_CA_CODE_LENGTH_CHIPS = 511.0

  GLONASS L1 C/A code length [chips].
- constexpr double GLONASS_L1_CA_CODE_PERIOD_S = 0.001

  GLONASS L1 C/A code period [seconds].
- constexpr double GLONASS_L1_CA_SYMBOL_PERIOD_S = 0.01

  GLONASS L1 C/A symbol period [seconds].
- constexpr double GLONASS_L1_CA_CHIP_PERIOD_S = 1.9569e-06

  GLONASS L1 C/A chip period [seconds].
- constexpr double GLONASS_L1_CA_SYMBOL_RATE_BPS = 1000.0
- constexpr uint32_t GLONASS_L1_CA_CODE_PERIOD_MS = 1
- constexpr uint32_t GLONASS_L2_CA_CODE_PERIOD_MS = 1
- constexpr int32_t GLONASS_CA_NBR_SATS = 24
- constexpr int32_t GLONASS_L1_CA_HISTORY_DEEP = 100
- constexpr char GLONASS_GNAV_PREAMBLE_STR [301] = "1111111111111111111111111111111111111111111111111111111100
- constexpr double GLONASS_GNAV_PREAMBLE_DURATION_S = 0.300
- constexpr double GLONASS_GNAV_SECONDS_PER_WEEK = 604800.0
- constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_BITS = 30
- constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS = 300
- constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS = 2000
- constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_BITS = 200
- constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND = 50

  NAV message bit rate [bits/s].
- constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT = 10
- constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT = 10
- constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND = GLONASS_GNAV_TELEMETRY_RATE_B
  ∗ GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT

  NAV message bit rate [symbols/s].
- constexpr int32_t GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE = 2000
- constexpr int32_t GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE = 200

  Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
- constexpr int32_t GLONASS_GNAV_STRING_BITS = 85

  Number of bits per string in the GNAV message (85 data bits + 30 time mark bits) [bits].
- constexpr int32_t GLONASS_GNAV_HAMMING_CODE_BITS = 8

  Number of bits in hamming code sequence of GNAV message.
- constexpr int32_t GLONASS_GNAV_DATA_SYMBOLS = 1700
- constexpr double GLONASS_LEAP_SECONDS [19][7]

  Record of leap seconds definition for GLOT to GPST conversion and vice versa.
- const std::map< uint32_t, int32_t > GLONASS_PRN
- const std::vector< int32_t > GLONASS_GNAV_CRC_I_INDEX {9, 10, 12, 13, 15, 17, 19, 20, 22, 24, 26, 28, 30, 32, 34, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84}
- const std::vector< int32_t > GLONASS_GNAV_CRC_J_INDEX {9, 11, 12, 14, 15, 18, 19, 21, 22, 25, 26, 29, 30, 33, 34, 36, 37, 40, 41, 44, 45, 48, 49, 52, 53, 56, 57, 60, 61, 64, 65, 67, 68, 71, 72, 75, 76, 79, 80, 83, 84}
- const std::vector< int32_t > GLONASS_GNAV_CRC_K_INDEX {10, 11, 12, 16, 17, 18, 19, 23, 24, 25, 26, 31, 32, 33, 34, 38, 39, 40, 41, 46, 47, 48, 49, 54, 55, 56, 57, 62, 63, 64, 65, 69, 70, 71, 72, 77, 78, 79, 80, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_L_INDEX {13, 14, 15, 16, 17, 18, 19, 27, 28, 29, 30, 31, 32, 33, 34, 42, 43, 44, 45, 46, 47, 48, 49, 58, 59, 60, 61, 62, 63, 64, 65, 73, 74, 75, 76, 77, 78, 79, 80}
- const std::vector< int32_t > GLONASS_GNAV_CRC_M_INDEX {20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_N_INDEX {35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65}

- const std::vector< int32_t > GLONASS_GNAV_CRC_P_INDEX {66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_CRC_Q_INDEX {9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85}
- const std::vector< int32_t > GLONASS_GNAV_ECC_LOCATOR {0, 0, 1, 8, 2, 9, 10, 11, 3, 12, 13, 14, 15, 16, 17, 18, 4, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 5, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 6, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84}

13.734.1 Detailed Description

Defines system parameters for GLONASS L1 C/A signal and NAV data.

Note

File renamed from GLONASS_L1_CA.h to GLONASS_L1_L2_CA.h to accommodate GLO L2 addition

Author

Damian Miralles, 2017. dmiralles2009(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file GLONASS_L1_L2_CA.h.

13.735 GLONASS_L1_L2_CA.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file GLONASS_L1_L2_CA.h
00003  * \brief  Defines system parameters for GLONASS L1 C/A signal and NAV data
00004  * \note File renamed from GLONASS_L1_CA.h to GLONASS_L1_L2_CA.h to accommodate GLO L2 addition
00005  * \author Damian Miralles, 2017. dmiralles2009(at)gmail.com
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GLONASS_L1_L2_CA_H
00020 #define GNSS_SDR_GLONASS_L1_L2_CA_H
00021
00022 #include "gnss_frequencies.h"
00023 #include <cstdint>
00024 #include <map>
00025 #include <utility>  // std::pair
00026 #include <vector>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033
00034 // Physical constants
00035 constexpr double GLONASS_F_M_A = 0.35e9;                          //!< Gravitational constant of
      atmosphere [m^3/s^2]
00036 constexpr double GLONASS_SEMI_MAJOR_AXIS = 6378136;               //!< Semi-major axis of Earth [m]
00037 constexpr double GLONASS_FLATTENING = 1.0 / 29825784.0;           //!< Flattening parameter
00038 constexpr double GLONASS_GRAVITY = 97803284.0;                    //!< Equatorial acceleration of
      gravity [mGal]
00039 constexpr double GLONASS_GRAVITY_CORRECTION = 0.87;               //!< Correction to acceleration of
      gravity at sea-level due to Atmosphere[uGal]
00040 constexpr double GLONASS_J2 = 1082625.75e-9;                     //!< Second zonal harmonic of the
      geopotential
```

```
00041 constexpr double GLONASS_J4 = -2370.89e-9;                          //!< Fourth zonal harmonic of the
      geopotential
00042 constexpr double GLONASS_J6 = 6.08e-9;                              //!< Sixth zonal harmonic of the
      geopotential
00043 constexpr double GLONASS_J8 = 1.40e-11;                             //!< Eighth zonal harmonic of the
      geopotential
00044 constexpr double GLONASS_U0 = 62636861.4;                           //!< Normal potential at surface of
      common terrestrial ellipsoid [m^2/s^2]
00045 constexpr double GLONASS_C20 = -1082.63e-6;                         //!< Second zonal coefficient of
      spherical harmonic expansion
00046 constexpr double GLONASS_EARTH_RADIUS = 6378.136;                   //!< Equatorial radius of Earth
      [km]
00047 constexpr double GLONASS_EARTH_INCLINATION = 0.000409148809899e3;  //!< Mean inclination of ecliptic
      to equator (23 deg 26 min 33 sec) [rad]
00048
00049 constexpr double GLONASS_TAU_0 = -0.005835151531174e3;  //!< (-334 deg 19 min 46.40 sec) [rad];
00050 constexpr double GLONASS_TAU_1 = 0.071018041257371e3;   //!< (4069 deg 02 min 02.52 sec) [rad];
00051
00052 constexpr double GLONASS_MOON_Q0 = -0.001115184961435e3;          //!< (-63 deg 53 min 43.41 sec)
      [rad]
00053 constexpr double GLONASS_MOON_Q1 = 8.328691103668023e3;           //!< (477198 deg 50 min 56.79 sec)
      [rad]
00054 constexpr double GLONASS_MOON_OMEGA_0 = 0.004523601514852e3;      //!< (259 deg 10 min 59.79 sec)
      [rad]
00055 constexpr double GLONASS_MOON_OMEGA_1 = -0.033757146246552e3;     //!< (-1934 deg 08 min 31.23 sec)
      [rad]
00056 constexpr double GLONASS_MOON_GM = 4902.835;                       //!< Lunar gravitational constant
      [km^3/s^2]
00057 constexpr double GLONASS_MOON_SEMI_MAJOR_AXIS = 3.84385243e5;     //!< Semi-major axis of lunar orbit
      [km];
00058 constexpr double GLONASS_MOON_ECCENTRICITY = 0.054900489;         //!< Eccentricity of lunar orbit
00059 constexpr double GLONASS_MOON_INCLINATION = 0.000089803977407e3;  //!< Inclination of lunar orbit to
      ecliptic plane (5 deg 08 min 43.4 sec) [rad]
00060
00061 constexpr double GLONASS_SUN_OMEGA = 0.004908229466869e3;  //!< TODO What is this operation in the
      seconds with T?(281 deg 13 min 15.0 + 6189.03 x T sec) [rad]
00062 constexpr double GLONASS_SUN_Q0 = 0.006256583774423e3;     //!< (358 deg 28 min 33.04 sec) [rad]
00063 constexpr double GLONASS_SUN_Q1 = 0e3;                     //!< TODO Why is the value greater than
      60?(129596579.10 sec) [rad]
00064 constexpr double GLONASS_SUN_GM = 0.1325263e12;            //!< Solar gravitational constant
      [km^3/s^2]
00065 constexpr double GLONASS_SUN_SEMI_MAJOR_AXIS = 1.49598e8;  //!< Semi-major axis of solar orbit [km];
00066 constexpr double GLONASS_SUN_ECCENTRICITY = 0.016719;       //!< Eccentricity of solar orbit
00067
00068 constexpr double GLONASS_L2_CA_FREQ_HZ = FREQ2_GLO;         //!< L2 [Hz]
00069 constexpr double GLONASS_L2_CA_DFREQ_HZ = DFRQ2_GLO;        //!< Freq Bias for GLONASS L1 [Hz]
00070 constexpr double GLONASS_L2_CA_CODE_RATE_CPS = 0.511e6;     //!< GLONASS L1 C/A code rate [chips/s]
00071 constexpr double GLONASS_L2_CA_CODE_LENGTH_CHIPS = 511.0;   //!< GLONASS L1 C/A code length [chips]
00072 constexpr double GLONASS_L2_CA_CODE_PERIOD_S = 0.001;       //!< GLONASS L1 C/A code period [seconds]
00073 constexpr double GLONASS_L2_CA_SYMBOL_PERIOD_S = 0.01;      //!< GLONASS L1 C/A symbol period
      [seconds]
00074 constexpr double GLONASS_L2_CA_CHIP_PERIOD_S = 1.9569e-06;  //!< GLONASS L1 C/A chip period [seconds]
00075 constexpr double GLONASS_L2_CA_SYMBOL_RATE_BPS = 1000.0;
00076
00077 constexpr double GLONASS_L1_CA_FREQ_HZ = FREQ1_GLO;         //!< L1 [Hz]
00078 constexpr double GLONASS_L1_CA_DFREQ_HZ = DFRQ1_GLO;        //!< Freq Bias for GLONASS L1 [Hz]
00079 constexpr double GLONASS_L1_CA_CODE_RATE_CPS = 0.511e6;     //!< GLONASS L1 C/A code rate [chips/s]
00080 constexpr double GLONASS_L1_CA_CODE_LENGTH_CHIPS = 511.0;   //!< GLONASS L1 C/A code length [chips]
00081 constexpr double GLONASS_L1_CA_CODE_PERIOD_S = 0.001;       //!< GLONASS L1 C/A code period [seconds]
00082 constexpr double GLONASS_L1_CA_SYMBOL_PERIOD_S = 0.01;      //!< GLONASS L1 C/A symbol period
      [seconds]
00083 constexpr double GLONASS_L1_CA_CHIP_PERIOD_S = 1.9569e-06;  //!< GLONASS L1 C/A chip period [seconds]
00084 constexpr double GLONASS_L1_CA_SYMBOL_RATE_BPS = 1000.0;
00085
00086 constexpr uint32_t GLONASS_L1_CA_CODE_PERIOD_MS = 1;
00087 constexpr uint32_t GLONASS_L2_CA_CODE_PERIOD_MS = 1;
00088 constexpr int32_t GLONASS_CA_NBR_SATS = 24;  // STRING DATA WITHOUT PREAMBLE
00089
00090 // OBSERVABLE HISTORY DEEP FOR INTERPOLATION
00091 constexpr int32_t GLONASS_L1_CA_HISTORY_DEEP = 100;
00092
00093 // NAVIGATION MESSAGE DEMODULATION AND DECODING
00094 #define GLONASS_GNAV_PREAMBLE \
00095     {                         \
00096         1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0}
00097
00098 #define GLONASS_GNAV_PREAMBLE_SAMPLES \
00099     {                                 \
00100         1, 1, 1, 1, 1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, 1, -1, -1, 1, -1,
      1, 1, -1}
00101
00102 constexpr char GLONASS_GNAV_PREAMBLE_STR[301] =
      "111111111111111111111111111111111111111111111111111100000000000000000000000000000001111111111111111111100000000000111111111
00103 constexpr double GLONASS_GNAV_PREAMBLE_DURATION_S = 0.300;
00104 constexpr double GLONASS_GNAV_SECONDS_PER_WEEK = 604800.0;
00105 constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_BITS = 30;
00106 constexpr int32_t GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS = 300;
```

```
00107 constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS = 2000;
00108 constexpr int32_t GLONASS_GNAV_PREAMBLE_PERIOD_BITS = 200;
00109 constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND = 50;  //!< NAV message bit rate [bits/s]
00110 constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT = 10;
00111 constexpr int32_t GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT = 10;
00112 constexpr int32_t GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND = GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND
      * GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT;  //!< NAV message bit rate [symbols/s]
00113 constexpr int32_t GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE = 2000;
00114 constexpr int32_t GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE = 200;  //!< Number of bits per string in
      the GNAV message (85 data bits + 30 time mark bits) [bits]
00115 constexpr int32_t GLONASS_GNAV_STRING_BITS = 85;                    //!< Number of bits per string in
      the GNAV message (85 data bits + 30 time mark bits) [bits]
00116 constexpr int32_t GLONASS_GNAV_HAMMING_CODE_BITS = 8;               //!< Number of bits in hamming
      code sequence of GNAV message
00117 constexpr int32_t GLONASS_GNAV_DATA_SYMBOLS = 1700;                 // STRING DATA WITHOUT PREAMBLE
00118
00119 /*!
00120  * \brief Record of leap seconds definition for GLOT to GPST conversion and vice versa
00121  * \details Each entry is defined by an array of 7 elements consisting of
      yr,month,day,hr,min,sec,utc-gpst
00122  * \note Ideally should use leap seconds definitions of rtklib
00123  */
00124 constexpr double GLONASS_LEAP_SECONDS[19][7] = {
00125     {2017, 1, 1, 0, 0, 0, -18},
00126     {2015, 7, 1, 0, 0, 0, -17},
00127     {2012, 7, 1, 0, 0, 0, -16},
00128     {2009, 1, 1, 0, 0, 0, -15},
00129     {2006, 1, 1, 0, 0, 0, -14},
00130     {1999, 1, 1, 0, 0, 0, -13},
00131     {1997, 7, 1, 0, 0, 0, -12},
00132     {1996, 1, 1, 0, 0, 0, -11},
00133     {1994, 7, 1, 0, 0, 0, -10},
00134     {1993, 7, 1, 0, 0, 0, -9},
00135     {1992, 7, 1, 0, 0, 0, -8},
00136     {1991, 1, 1, 0, 0, 0, -7},
00137     {1990, 1, 1, 0, 0, 0, -6},
00138     {1988, 1, 1, 0, 0, 0, -5},
00139     {1985, 7, 1, 0, 0, 0, -4},
00140     {1983, 7, 1, 0, 0, 0, -3},
00141     {1982, 7, 1, 0, 0, 0, -2},
00142     {1981, 7, 1, 0, 0, 0, -1},
00143     {}};
00144
00145 // GLONASS RINEX satellite numbers are slot numbers. This table maps Rnn slot
00146 // numbers to the corresponding FDMA frequency channel.
00147 const std::map<uint32_t, int32_t> GLONASS_PRN = {
00148     {
00149         0,
00150         8,
00151     },  // For test
00152     {
00153         1,
00154         1,
00155     },  // Plane 1
00156     {
00157         2,
00158         -4,
00159     },  // Plane 1
00160     {
00161         3,
00162         5,
00163     },  // Plane 1
00164     {
00165         4,
00166         6,
00167     },  // Plane 1
00168     {
00169         5,
00170         1,
00171     },  // Plane 1
00172     {
00173         6,
00174         -4,
00175     },  // Plane 1
00176     {
00177         7,
00178         5,
00179     },  // Plane 1
00180     {
00181         8,
00182         6,
00183     },  // Plane 1
00184     {
00185         9,
00186         -2,
00187     },  // Plane 2
00188     {
```

```
00189         10,
00190         -7,
00191     },  // Plane 2
00192     {
00193         11,
00194         0,
00195     },  // Plane 2
00196     {
00197         12,
00198         -1,
00199     },  // Plane 2
00200     {
00201         13,
00202         -2,
00203     },  // Plane 2
00204     {
00205         14,
00206         -7,
00207     },  // Plane 2
00208     {
00209         15,
00210         0,
00211     },  // Plane 2
00212     {
00213         16,
00214         -1,
00215     },  // Plane 2
00216     {
00217         17,
00218         4,
00219     },  // Plane 3
00220     {
00221         18,
00222         -3,
00223     },  // Plane 3
00224     {
00225         19,
00226         3,
00227     },  // Plane 3
00228     {
00229         20,
00230         2,
00231     },  // Plane 3
00232     {
00233         21,
00234         4,
00235     },  // Plane 3
00236     {
00237         22,
00238         -3,
00239     },  // Plane 3
00240     {
00241         23,
00242         3,
00243     },  // Plane 3
00244     {
00245         24,
00246         2,
00247     },  // Plane 3
00248     {
00249         26,
00250         -6,
00251     },
00252     {
00253         27,
00254         -5,
00255     },
00256     {
00257         28,
00258         7,
00259     }};
00260
00261 const std::vector<int32_t> GLONASS_GNAV_CRC_I_INDEX{9, 10, 12, 13, 15, 17, 19, 20, 22, 24, 26, 28, 30,
      32, 34, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 66, 68, 70, 72, 74, 76, 78,
      80, 82, 84};
00262 const std::vector<int32_t> GLONASS_GNAV_CRC_J_INDEX{9, 11, 12, 14, 15, 18, 19, 21, 22, 25, 26, 29, 30,
      33, 34, 36, 37, 40, 41, 44, 45, 48, 49, 52, 53, 56, 57, 60, 61, 64, 65, 67, 68, 71, 72, 75, 76, 79,
      80, 83, 84};
00263 const std::vector<int32_t> GLONASS_GNAV_CRC_K_INDEX{10, 11, 12, 16, 17, 18, 19, 23, 24, 25, 26, 31,
      32, 33, 34, 38, 39, 40, 41, 46, 47, 48, 49, 54, 55, 56, 57, 62, 63, 64, 65, 69, 70, 71, 72, 77, 78,
      79, 80, 85};
00264 const std::vector<int32_t> GLONASS_GNAV_CRC_L_INDEX{13, 14, 15, 16, 17, 18, 19, 27, 28, 29, 30, 31,
      32, 33, 34, 42, 43, 44, 45, 46, 47, 48, 49, 58, 59, 60, 61, 62, 63, 64, 65, 73, 74, 75, 76, 77, 78,
      79, 80};
00265 const std::vector<int32_t> GLONASS_GNAV_CRC_M_INDEX{20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31,
      32, 33, 34, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 81, 82, 83, 84, 85};
00266 const std::vector<int32_t> GLONASS_GNAV_CRC_N_INDEX{35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46,
```

```
      47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65};
00267 const std::vector<int32_t> GLONASS_GNAV_CRC_P_INDEX{66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,
      78, 79, 80, 81, 82, 83, 84, 85};
00268 const std::vector<int32_t> GLONASS_GNAV_CRC_Q_INDEX{9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
      22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46,
      47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71,
      72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85};
00269 const std::vector<int32_t> GLONASS_GNAV_ECC_LOCATOR{0, 0, 1, 8, 2, 9, 10, 11, 3, 12, 13, 14, 15, 16,
      17, 18, 4, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 5, 34, 35, 36, 37, 38, 39, 40,
      41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 6, 65,
      66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84};
00270
00271 // GLONASS GNAV NAVIGATION MESSAGE STRUCTURE
00272 // NAVIGATION MESSAGE FIELDS POSITIONS
00273
00274 // FRAME 1-4
00275 // COMMON FIELDS
00276 const std::vector<std::pair<int32_t, int32_t» STRING_ID({{2, 4}});
00277 const std::vector<std::pair<int32_t, int32_t» KX({{78, 8}});
00278 // STRING 1
00279 const std::vector<std::pair<int32_t, int32_t» P1({{8, 2}});
00280 const std::vector<std::pair<int32_t, int32_t» T_K_HR({{10, 5}});
00281 const std::vector<std::pair<int32_t, int32_t» T_K_MIN({{15, 6}});
00282 const std::vector<std::pair<int32_t, int32_t» T_K_SEC({{21, 1}});
00283 const std::vector<std::pair<int32_t, int32_t» X_N_DOT({{22, 24}});
00284 const std::vector<std::pair<int32_t, int32_t» X_N_DOT_DOT({{46, 5}});
00285 const std::vector<std::pair<int32_t, int32_t» X_N({{51, 27}});
00286
00287 // STRING 2
00288 const std::vector<std::pair<int32_t, int32_t» B_N({{6, 3}});
00289 const std::vector<std::pair<int32_t, int32_t» P2({{9, 1}});
00290 const std::vector<std::pair<int32_t, int32_t» T_B({{10, 7}});
00291 const std::vector<std::pair<int32_t, int32_t» Y_N_DOT({{22, 24}});
00292 const std::vector<std::pair<int32_t, int32_t» Y_N_DOT_DOT({{46, 5}});
00293 const std::vector<std::pair<int32_t, int32_t» Y_N({{51, 27}});
00294
00295 // STRING 3
00296 const std::vector<std::pair<int32_t, int32_t» P3({{6, 1}});
00297 const std::vector<std::pair<int32_t, int32_t» GAMMA_N({{7, 11}});
00298 const std::vector<std::pair<int32_t, int32_t» P({{19, 2}});
00299 const std::vector<std::pair<int32_t, int32_t» EPH_L_N({{21, 1}});
00300 const std::vector<std::pair<int32_t, int32_t» Z_N_DOT({{22, 24}});
00301 const std::vector<std::pair<int32_t, int32_t» Z_N_DOT_DOT({{46, 5}});
00302 const std::vector<std::pair<int32_t, int32_t» Z_N({{51, 27}});
00303
00304 // STRING 4
00305 const std::vector<std::pair<int32_t, int32_t» TAU_N({{6, 22}});
00306 const std::vector<std::pair<int32_t, int32_t» DELTA_TAU_N({{28, 5}});
00307 const std::vector<std::pair<int32_t, int32_t» E_N({{33, 5}});
00308 const std::vector<std::pair<int32_t, int32_t» P4({{52, 1}});
00309 const std::vector<std::pair<int32_t, int32_t» F_T({{53, 4}});
00310 const std::vector<std::pair<int32_t, int32_t» N_T({{60, 11}});
00311 const std::vector<std::pair<int32_t, int32_t» N({{71, 5}});
00312 const std::vector<std::pair<int32_t, int32_t» M({{76, 2}});
00313
00314 // STRING 5
00315 const std::vector<std::pair<int32_t, int32_t» DAY_NUMBER_A({{6, 11}});
00316 const std::vector<std::pair<int32_t, int32_t» TAU_C({{17, 32}});
00317 const std::vector<std::pair<int32_t, int32_t» N_4({{50, 5}});
00318 const std::vector<std::pair<int32_t, int32_t» TAU_GPS({{55, 22}});
00319 const std::vector<std::pair<int32_t, int32_t» ALM_L_N({{77, 1}});
00320
00321 // STRING 6, 8, 10, 12, 14
00322 const std::vector<std::pair<int32_t, int32_t» C_N({{6, 1}});
00323 const std::vector<std::pair<int32_t, int32_t» M_N_A({{7, 2}});
00324 const std::vector<std::pair<int32_t, int32_t» N_A({{9, 5}});
00325 const std::vector<std::pair<int32_t, int32_t» TAU_N_A({{14, 10}});
00326 const std::vector<std::pair<int32_t, int32_t» LAMBDA_N_A({{24, 21}});
00327 const std::vector<std::pair<int32_t, int32_t» DELTA_I_N_A({{45, 18}});
00328 const std::vector<std::pair<int32_t, int32_t» EPSILON_N_A({{63, 15}});
00329
00330 // STRING 7, 9, 11, 13, 15
00331 const std::vector<std::pair<int32_t, int32_t» OMEGA_N_A({{6, 16}});
00332 const std::vector<std::pair<int32_t, int32_t» T_LAMBDA_N_A({{22, 21}});
00333 const std::vector<std::pair<int32_t, int32_t» DELTA_T_N_A({{43, 22}});
00334 const std::vector<std::pair<int32_t, int32_t» DELTA_T_DOT_N_A({{65, 7}});
00335 const std::vector<std::pair<int32_t, int32_t» H_N_A({{72, 5}});
00336
00337 // STRING 14 FRAME 5
00338 const std::vector<std::pair<int32_t, int32_t» B1({{6, 11}});
00339 const std::vector<std::pair<int32_t, int32_t» B2({{17, 10}});
00340 const std::vector<std::pair<int32_t, int32_t» KP({{27, 2}});
00341
00342
00343 /** \} */
00344 /** \} */
00345 #endif  // GNSS_SDR_GLONASS_L1_L2_CA_H
```

13.736 gnss_almanac.h File Reference

Base class for GNSS almanac storage.

```
#include <array>
#include <cstdint>
```

Classes

- class Gnss_Almanac

  Base class for GNSS almanac storage.

13.736.1 Detailed Description

Base class for GNSS almanac storage.

Author

Carles Fernandez, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_almanac.h.

13.737 gnss_almanac.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_almanac.h
00003  * \brief Base class for GNSS almanac storage
00004  * \author Carles Fernandez, 2021. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_ALMANAC_H
00019 #define GNSS_SDR_GNSS_ALMANAC_H
00020
00021 #include <array>
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief Base class for GNSS almanac storage
00032  */
00033 class Gnss_Almanac
00034 {
00035 public:
00036     /*!
00037      * Default constructor
00038      */
00039     Gnss_Almanac() = default;
00040
00041     /*!
00042      * \brief Computes prediction of the Doppler shift for a given time and receiver's position and
      velocity.
00043      * \f[
00044      * f_{d} = - \mathbf{v} \frac{\mathbf{x}^{T}}{\left| \mathbf{x} \right| } \frac{f_{L}}{c}
00045      * \f]
00046      * where:
00047      * \f[
```

```
00048      * \mathbf{v} = \mathbf{v}_{sat} - \mathbf{v}_{rx}
00049      * \f]
00050      * \f[
00051      * \mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{rx}
00052      * \f]
00053      * \f[
00054      * \left| \mathbf{x} \right| = \sqrt{\mathbf{x}\mathbf{x}^{T}}
00055      * \f]
00056      *
00057      * @param[in] rx_time_s Time of Week in seconds
00058      * @param[in] lat Receiver's latitude in degrees
00059      * @param[in] lon Receiver's longitude in degrees
00060      * @param[in] h   Receiver's height in meters
00061      * @param[in] ve  Receiver's velocity in the East direction [m/s]
00062      * @param[in] vn  Receiver's velocity in the North direction [m/s]
00063      * @param[in] vu  Receiver's velocity in the Up direction [m/s]
00064      * @param[in] band Signal band for which the Doppler will be computed
00065      *                 (1: L1 C/A, E1B, BI1; 2: L2C, BI2; 3: BI3; 5: L5/E5a; 6: E6B; 7: E5b; 8:
      E5a+E5b)
00066      */
00067     double predicted_doppler(double rx_time_s,
00068         double lat,
00069         double lon,
00070         double h,
00071         double ve,
00072         double vn,
00073         double vu,
00074         int band) const;
00075
00076     /*!
00077      * \brief Computes satellite Position and Velocity, in ECEF, for a given time (expressed in
      seconds of week)
00078      */
00079     void satellitePosVelComputation(double transmitTime, std::array<double, 7>& pos_vel_dtr) const;
00080
00081     uint32_t PRN{};      //!< SV PRN NUMBER
00082     double delta_i{};    //!< Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles)
00083     int32_t toa{};       //!< Almanac data reference time of week [s]
00084     int32_t WNa{};       //!< Almanac week number
00085     double M_0{};        //!< Mean Anomaly at Reference Time [semi-circles]
00086     double ecc{};        //!< Eccentricity [dimensionless]
00087     double sqrtA{};      //!< Square Root of the Semi-Major Axis [sqrt(m)]
00088     double OMEGA_0{};    //!< Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
00089     double omega{};      //!< Argument of Perigee [semi-cicles]
00090     double OMEGAdot{};   //!< Rate of Right Ascension [semi-circles/s]
00091     double af0{};        //!< Coefficient 0 of code phase offset model [s]
00092     double af1{};        //!< Coefficient 1 of code phase offset model [s/s]
00093
00094     //! Loaded by the AGNSS/SUPL startup bootstrap rather than decoded from the
00095     //! broadcast. Runtime-only provenance flag, intentionally not serialized.
00096     bool from_startup_load{false};
00097
00098 protected:
00099     char System{};  //!< Character ID of the GNSS system. 'G': GPS. 'E': Galileo. 'C': BeiDou. 'J':
      QZSS
00100 private:
00101     double check_t(double time) const;
00102 };
00103
00104
00105 /** \} */
00106 /** \} */
00107 #endif  // GNSS_SDR_GNSS_ALMANAC_H
```

13.738 gnss_ephemeris.h File Reference

Base class for GNSS Ephemeris.

```
#include <array>
#include <cstdint>
```

Classes

- class Gnss_Ephemeris

  Base class for GNSS ephemeris storage.

13.738.1 Detailed Description

Base class for GNSS Ephemeris.

Author

Carles Fernandez, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_ephemeris.h.

13.739 gnss_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_ephemeris.h
00003  * \brief Base class for GNSS Ephemeris
00004  * \author Carles Fernandez, 2021. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GNSS_EPHEMERIS_H
00020 #define GNSS_SDR_GNSS_EPHEMERIS_H
00021
00022 #include <array>
00023 #include <cstdint>
00024
00025 /*!
00026  * \brief Base class for GNSS ephemeris storage
00027  */
00028 class Gnss_Ephemeris
00029 {
00030 public:
00031     Gnss_Ephemeris() = default;
00032
00033     /*!
00034      * \brief Sets (\a satClkDrift) and (\a dtr), and returns the clock drift in
00035      * seconds according to the User Algorithm for SV Clock Correction
00036      * (IS-GPS-200M, 20.3.3.3.3.1, and Galileo OS SIS ICD, 5.1.4).
00037      */
00038     double sv_clock_drift(double transmitTime);
00039
00040     /*!
00041      * \brief Computes prediction of the Doppler shift for a given time and receiver's position and
      velocity.
00042      * \f[
00043      * f_{d} = - \mathbf{v} \frac{\mathbf{x}^{T}}{\left| \mathbf{x} \right| } \frac{f_{L}}{c}
00044      * \f]
00045      * where:
00046      * \f[
00047      * \mathbf{v} = \mathbf{v}_{sat} - \mathbf{v}_{rx}
00048      * \f]
00049      * \f[
00050      * \mathbf{x} = \mathbf{x}_{sat} - \mathbf{x}_{rx}
00051      * \f]
00052      * \f[
00053      * \left| \mathbf{x} \right| = \sqrt{\mathbf{x}\mathbf{x}^{T}}
00054      * \f]
00055      *
00056      * @param[in] rx_time_s Time of Week in seconds
00057      * @param[in] lat Receiver's latitude in degrees
00058      * @param[in] lon Receiver's longitude in degrees
00059      * @param[in] h   Receiver's height in meters
00060      * @param[in] ve  Receiver's velocity in the East direction [m/s]
00061      * @param[in] vn  Receiver's velocity in the North direction [m/s]
00062      * @param[in] vu  Receiver's velocity in the Up direction [m/s]
00063      * @param[in] band Signal band for which the Doppler will be computed
00064      *                 (1: L1 C/A, E1B, BI1; 2: L2C, BI2; 3: BI3; 5: L5/E5a; 6: E6B; 7: E5b; 8:
      E5a+E5b)
00065      */
```

```
00066     double predicted_doppler(double rx_time_s, double lat, double lon, double h, double ve, double vn,
      double vu, int band) const;
00067
00068     void satellitePosition(double transmitTime);  //!< Computes the ECEF SV coordinates and ECEF
      velocity
00069
00070     uint32_t PRN{};       //!< SV ID
00071     double M_0{};         //!< Mean anomaly at reference time [rad]
00072     double Adot{};        //!< Change rate in semi-major axis (CNAV)
00073     double delta_ndot{};  //!< Rate of mean motion difference from computed value (CNAV)
00074     double delta_n{};     //!< Mean motion difference from computed value [rad/sec]
00075     double ecc{};         //!< Eccentricity
00076     double sqrtA{};       //!< Square root of the semi-major axis [meters^1/2]
00077     double A0{};          //!< Semi-major axis [meters] (B-CNAV1); unused when sqrtA is set
00078     double OMEGA_0{};     //!< Longitude of ascending node of orbital plane at weekly epoch [rad]
00079     double i_0{};         //!< Inclination angle at reference time [rad]
00080     double omega{};       //!< Argument of perigee [rad]
00081     double OMEGAdot{};    //!< Rate of right ascension [rad/sec]
00082     double idot{};        //!< Rate of inclination angle [rad/sec]
00083     double Cuc{};         //!< Amplitude of the cosine harmonic correction term to the argument of
      latitude [rad]
00084     double Cus{};         //!< Amplitude of the sine harmonic correction term to the argument of
      latitude [rad]
00085     double Crc{};         //!< Amplitude of the cosine harmonic correction term to the orbit radius
      [meters]
00086     double Crs{};         //!< Amplitude of the sine harmonic correction term to the orbit radius
      [meters]
00087     double Cic{};         //!< Amplitude of the cosine harmonic correction term to the angle of
      inclination [rad]
00088     double Cis{};         //!< Amplitude of the sine harmonic correction term to the angle of
      inclination [rad]
00089     int32_t toe{};        //!< Ephemeris reference time [s]
00090
00091     // Clock correction parameters
00092     int32_t toc{};  //!< Clock correction data reference Time of Week [sec]
00093     double af0{};   //!< SV clock bias correction coefficient [s]
00094     double af1{};   //!< SV clock drift correction coefficient [s/s]
00095     double af2{};   //!< SV clock drift rate correction coefficient [s/s^2]
00096
00097     double satClkDrift{};  //!< SV clock drift
00098     double dtr{};          //!< Relativistic clock correction term
00099
00100     // Time
00101     int32_t WN{};   //!< Week number
00102     int32_t tow{};  //!< Time of Week
00103
00104     // satellite positions
00105     double satpos_X{};  //!< Earth-fixed coordinate x of the satellite [m]. Intersection of the IERS
      Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.
00106     double satpos_Y{};  //!< Earth-fixed coordinate y of the satellite [m]. Completes a right-handed,
      Earth-Centered, Earth-Fixed orthogonal coordinate system.
00107     double satpos_Z{};  //!< Earth-fixed coordinate z of the satellite [m]. The direction of the IERS
      (International Earth Rotation and Reference Systems Service) Reference Pole (IRP).
00108
00109     // Satellite velocity
00110     double satvel_X{};  //!< Earth-fixed velocity coordinate x of the satellite [m]
00111     double satvel_Y{};  //!< Earth-fixed velocity coordinate y of the satellite [m]
00112     double satvel_Z{};  //!< Earth-fixed velocity coordinate z of the satellite [m]
00113
00114 protected:
00115     char System{};  //!< Character ID of the GNSS system. 'G': GPS.  'E': Galileo.  'C': BeiDou
00116
00117 private:
00118     void satellitePosVelComputation(double transmitTime, std::array<double, 7>& pos_vel_dtr) const;
00119     double check_t(double time) const;
00120     double sv_clock_relativistic_term(double transmitTime) const;
00121 };
00122
00123 #endif  // GNSS_SDR_GNSS_EPHEMERIS_H
```

13.740 gnss_frequencies.h File Reference

GNSS Frequencies.

```
#include <string>
#include <unordered_map>
```

Variables

- constexpr double FREQ1 = 1.57542e9

L1/E1 frequency (Hz).

- constexpr double FREQ2 = 1.22760e9

  L2 frequency (Hz).
- constexpr double FREQ5 = 1.17645e9

  L5/E5a frequency (Hz).
- constexpr double FREQ6 = 1.27875e9

  E6/LEX frequency (Hz).
- constexpr double FREQ7 = 1.20714e9

  E5b frequency (Hz).
- constexpr double FREQ8 = 1.191795e9

  E5a+b frequency (Hz).
- constexpr double FREQ9 = 2.492028e9

  S frequency (Hz).
- constexpr double FREQ1_GLO = 1.60200e9

  GLONASS G1 base frequency (Hz).
- constexpr double DFRQ1_GLO = 0.56250e6

  GLONASS G1 bias frequency (Hz/n).
- constexpr double FREQ2_GLO = 1.24600e9

  GLONASS G2 base frequency (Hz).
- constexpr double DFRQ2_GLO = 0.43750e6

  GLONASS G2 bias frequency (Hz/n).
- constexpr double FREQ3_GLO = 1.202025e9

  GLONASS G3 frequency (Hz).
- constexpr double FREQ1_BDS = 1.561098e9

  BeiDou B1 frequency (Hz).
- constexpr double FREQ2_BDS = 1.20714e9

  BeiDou B2 frequency (Hz).
- constexpr double FREQ3_BDS = 1.26852e9

  BeiDou B3 frequency (Hz).
- const std::unordered_map< std::string, double > SIGNAL_FREQ_MAP

13.740.1 Detailed Description

GNSS Frequencies.

Author

Carles Fernandez, 2017. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_frequencies.h.

13.741 gnss_frequencies.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_frequencies.h
00003  * \brief  GNSS Frequencies
00004  * \author Carles Fernandez, 2017. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
```

```
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GNSS_FREQUENCIES_H
00020 #define GNSS_SDR_GNSS_FREQUENCIES_H
00021
00022 #include <string>
00023 #include <unordered_map>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 constexpr double FREQ1 = 1.57542e9;        //!<  L1/E1  frequency (Hz)
00032 constexpr double FREQ2 = 1.22760e9;        //!<  L2     frequency (Hz)
00033 constexpr double FREQ5 = 1.17645e9;        //!<  L5/E5a frequency (Hz)
00034 constexpr double FREQ6 = 1.27875e9;        //!<  E6/LEX frequency (Hz)
00035 constexpr double FREQ7 = 1.20714e9;        //!<  E5b    frequency (Hz)
00036 constexpr double FREQ8 = 1.191795e9;       //!<  E5a+b  frequency (Hz)
00037 constexpr double FREQ9 = 2.492028e9;       //!<  S      frequency (Hz)
00038 constexpr double FREQ1_GLO = 1.60200e9;    //!<  GLONASS G1 base frequency (Hz)
00039 constexpr double DFRQ1_GLO = 0.56250e6;    //!<  GLONASS G1 bias frequency (Hz/n)
00040 constexpr double FREQ2_GLO = 1.24600e9;    //!<  GLONASS G2 base frequency (Hz)
00041 constexpr double DFRQ2_GLO = 0.43750e6;    //!<  GLONASS G2 bias frequency (Hz/n)
00042 constexpr double FREQ3_GLO = 1.202025e9;   //!<  GLONASS G3 frequency (Hz)
00043 constexpr double FREQ1_BDS = 1.561098e9;   //!<  BeiDou B1 frequency (Hz)
00044 constexpr double FREQ2_BDS = 1.20714e9;    //!<  BeiDou B2 frequency (Hz)
00045 constexpr double FREQ3_BDS = 1.26852e9;    //!<  BeiDou B3 frequency (Hz)
00046
00047 const std::unordered_map<std::string, double> SIGNAL_FREQ_MAP = {
00048     {"1C", FREQ1},
00049     {"2S", FREQ2},
00050     {"L5", FREQ5},
00051     {"1B", FREQ1},
00052     {"5X", FREQ5},
00053     {"E6", FREQ6},
00054     {"7X", FREQ7},
00055     {"1G", FREQ1_GLO},
00056     {"2G", FREQ2_GLO},
00057     {"B1", FREQ1_BDS},
00058     {"B2", FREQ2_BDS},
00059     {"5D", FREQ5},
00060     {"B3", FREQ3_BDS},
00061     {"1D", FREQ1},
00062     {"J1", FREQ1},
00063     {"J5", FREQ5},
00064 };
00065
00066
00067 /** \} */
00068 /** \} */
00069 #endif  // GNSS_SDR_GNSS_FREQUENCIES_H
```

13.742 gnss_obs_codes.h File Reference

GNSS Observable codes.

`#include <cstdint>`

Variables

- constexpr uint32_t CODE_NONE = 0

  obs code: none or unknown
- constexpr uint32_t CODE_L1C = 1

  obs code: L1C/A,G1C/A,E1C (GPS,GLO,GAL,QZS,SBS)
- constexpr uint32_t CODE_L1P = 2

  obs code: L1P,G1P (GPS,GLO)
- constexpr uint32_t CODE_L1W = 3

  obs code: L1 Z-track (GPS)
- constexpr uint32_t CODE_L1Y = 4

  obs code: L1Y (GPS)

- constexpr uint32_t CODE_L1M = 5
  obs code: L1M (GPS)
- constexpr uint32_t CODE_L1N = 6
  obs code: L1codeless (GPS)
- constexpr uint32_t CODE_L1S = 7
  obs code: L1C(D) (GPS,QZS)
- constexpr uint32_t CODE_L1L = 8
  obs code: L1C(P) (GPS,QZS)
- constexpr uint32_t CODE_L1E = 9
  (not used)
- constexpr uint32_t CODE_L1A = 10
  obs code: E1A (GAL)
- constexpr uint32_t CODE_L1B = 11
  obs code: E1B (GAL)
- constexpr uint32_t CODE_L1X = 12
  obs code: E1B+C,L1C(D+P) (GAL,QZS)
- constexpr uint32_t CODE_L1Z = 13
  obs code: E1A+B+C,L1SAIF (GAL,QZS)
- constexpr uint32_t CODE_L2C = 14
  obs code: L2C/A,G1C/A (GPS,GLO)
- constexpr uint32_t CODE_L2D = 15
  obs code: L2 L1C/A-(P2-P1) (GPS)
- constexpr uint32_t CODE_L2S = 16
  obs code: L2C(M) (GPS,QZS)
- constexpr uint32_t CODE_L2L = 17
  obs code: L2C(L) (GPS,QZS)
- constexpr uint32_t CODE_L2X = 18
  obs code: L2C(M+L),B1I+Q (GPS,QZS,BDS)
- constexpr uint32_t CODE_L2P = 19
  obs code: L2P,G2P (GPS,GLO)
- constexpr uint32_t CODE_L2W = 20
  obs code: L2 Z-track (GPS)
- constexpr uint32_t CODE_L2Y = 21
  obs code: L2Y (GPS)
- constexpr uint32_t CODE_L2M = 22
  obs code: L2M (GPS)
- constexpr uint32_t CODE_L2N = 23
  obs code: L2codeless (GPS)
- constexpr uint32_t CODE_L5I = 24
  obs code: L5/E5aI (GPS,GAL,QZS,SBS)
- constexpr uint32_t CODE_L5Q = 25
  obs code: L5/E5aQ (GPS,GAL,QZS,SBS)
- constexpr uint32_t CODE_L5X = 26
  obs code: L5/E5aI+Q/L5B+C (GPS,GAL,QZS,IRN,SBS)
- constexpr uint32_t CODE_L7I = 27
  obs code: E5bI,B2I (GAL,BDS)
- constexpr uint32_t CODE_L7Q = 28
  obs code: E5bQ,B2Q (GAL,BDS)
- constexpr uint32_t CODE_L7X = 29
  obs code: E5bI+Q,B2I+Q (GAL,BDS)
- constexpr uint32_t CODE_L6A = 30

obs code: E6A (GAL)

- constexpr uint32_t CODE_L6B = 31

  obs code: E6B (GAL)
- constexpr uint32_t CODE_L6C = 32

  obs code: E6C (GAL)
- constexpr uint32_t CODE_L6X = 33

  obs code: E6B+C,LEXS+L,B3I+Q (GAL,QZS,BDS)
- constexpr uint32_t CODE_L6Z = 34

  obs code: E6A+B+C (GAL)
- constexpr uint32_t CODE_L6S = 35

  obs code: LEXS (QZS)
- constexpr uint32_t CODE_L6L = 36

  obs code: LEXL (QZS)
- constexpr uint32_t CODE_L8I = 37

  obs code: E5(a+b)I (GAL)
- constexpr uint32_t CODE_L8Q = 38

  obs code: E5(a+b)Q (GAL)
- constexpr uint32_t CODE_L8X = 39

  obs code: E5(a+b)I+Q (GAL)
- constexpr uint32_t CODE_L2I = 40

  obs code: B1I (BDS)
- constexpr uint32_t CODE_L2Q = 41

  obs code: B1Q (BDS)
- constexpr uint32_t CODE_L6I = 42

  obs code: B3I (BDS)
- constexpr uint32_t CODE_L6Q = 43

  obs code: B3Q (BDS)
- constexpr uint32_t CODE_L3I = 44

  obs code: G3I (GLO)
- constexpr uint32_t CODE_L3Q = 45

  obs code: G3Q (GLO)
- constexpr uint32_t CODE_L3X = 46

  obs code: G3I+Q (GLO)
- constexpr uint32_t CODE_L1I = 47

  obs code: B1I (BDS)
- constexpr uint32_t CODE_L1Q = 48

  obs code: B1Q (BDS)
- constexpr uint32_t CODE_L5A = 49

  obs code: L5A SPS (IRN)
- constexpr uint32_t CODE_L5B = 50

  obs code: L5B RS(D) (IRN)
- constexpr uint32_t CODE_L5C = 51

  obs code: L5C RS(P) (IRN)
- constexpr uint32_t CODE_L9A = 52

  obs code: SA SPS (IRN)
- constexpr uint32_t CODE_L9B = 53

  obs code: SB RS(D) (IRN)
- constexpr uint32_t CODE_L9C = 54

  obs code: SC RS(P) (IRN)
- constexpr uint32_t CODE_L9X = 55

  obs code: SB+C (IRN)

- constexpr uint32_t CODE_L1D = 56

  obs code: B1C data (BDS)
- constexpr uint32_t CODE_L5D = 57

  obs code: B2a data (BDS)
- constexpr int32_t MAXCODE = 57

  max number of obs code

13.742.1 Detailed Description

GNSS Observable codes.

Author

Carles Fernandez, 2017. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_obs_codes.h.

13.743 gnss_obs_codes.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_obs_codes.h
00003  * \brief  GNSS Observable codes
00004  * \author Carles Fernandez, 2017. cfernandez(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GNSS_OBS_CODES_H
00020 #define GNSS_SDR_GNSS_OBS_CODES_H
00021
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 constexpr uint32_t CODE_NONE = 0;  //!<   obs code: none or unknown
00031 constexpr uint32_t CODE_L1C = 1;   //!<   obs code: L1C/A,G1C/A,E1C (GPS,GLO,GAL,QZS,SBS)
00032 constexpr uint32_t CODE_L1P = 2;   //!<   obs code: L1P,G1P    (GPS,GLO)
00033 constexpr uint32_t CODE_L1W = 3;   //!<   obs code: L1 Z-track (GPS)
00034 constexpr uint32_t CODE_L1Y = 4;   //!<   obs code: L1Y        (GPS)
00035 constexpr uint32_t CODE_L1M = 5;   //!<   obs code: L1M        (GPS)
00036 constexpr uint32_t CODE_L1N = 6;   //!<   obs code: L1codeless (GPS)
00037 constexpr uint32_t CODE_L1S = 7;   //!<   obs code: L1C(D)     (GPS,QZS)
00038 constexpr uint32_t CODE_L1L = 8;   //!<   obs code: L1C(P)     (GPS,QZS)
00039 constexpr uint32_t CODE_L1E = 9;   //!<   (not used)
00040 constexpr uint32_t CODE_L1A = 10;  //!<   obs code: E1A        (GAL)
00041 constexpr uint32_t CODE_L1B = 11;  //!<   obs code: E1B        (GAL)
00042 constexpr uint32_t CODE_L1X = 12;  //!<   obs code: E1B+C,L1C(D+P) (GAL,QZS)
00043 constexpr uint32_t CODE_L1Z = 13;  //!<   obs code: E1A+B+C,L1SAIF (GAL,QZS)
00044 constexpr uint32_t CODE_L2C = 14;  //!<   obs code: L2C/A,G1C/A (GPS,GLO)
00045 constexpr uint32_t CODE_L2D = 15;  //!<   obs code: L2 L1C/A-(P2-P1) (GPS)
00046 constexpr uint32_t CODE_L2S = 16;  //!<   obs code: L2C(M)     (GPS,QZS)
00047 constexpr uint32_t CODE_L2L = 17;  //!<   obs code: L2C(L)     (GPS,QZS)
00048 constexpr uint32_t CODE_L2X = 18;  //!<   obs code: L2C(M+L),B1I+Q (GPS,QZS,BDS)
00049 constexpr uint32_t CODE_L2P = 19;  //!<   obs code: L2P,G2P    (GPS,GLO)
00050 constexpr uint32_t CODE_L2W = 20;  //!<   obs code: L2 Z-track (GPS)
00051 constexpr uint32_t CODE_L2Y = 21;  //!<   obs code: L2Y        (GPS)
00052 constexpr uint32_t CODE_L2M = 22;  //!<   obs code: L2M        (GPS)
00053 constexpr uint32_t CODE_L2N = 23;  //!<   obs code: L2codeless (GPS)
00054 constexpr uint32_t CODE_L5I = 24;  //!<   obs code: L5/E5aI    (GPS,GAL,QZS,SBS)
```

```
00055 constexpr uint32_t CODE_L5Q = 25;   //!<   obs code: L5/E5aQ     (GPS,GAL,QZS,SBS)
00056 constexpr uint32_t CODE_L5X = 26;   //!<   obs code: L5/E5aI+Q/L5B+C (GPS,GAL,QZS,IRN,SBS)
00057 constexpr uint32_t CODE_L7I = 27;   //!<   obs code: E5bI,B2I    (GAL,BDS)
00058 constexpr uint32_t CODE_L7Q = 28;   //!<   obs code: E5bQ,B2Q    (GAL,BDS)
00059 constexpr uint32_t CODE_L7X = 29;   //!<   obs code: E5bI+Q,B2I+Q (GAL,BDS)
00060 constexpr uint32_t CODE_L6A = 30;   //!<   obs code: E6A         (GAL)
00061 constexpr uint32_t CODE_L6B = 31;   //!<   obs code: E6B         (GAL)
00062 constexpr uint32_t CODE_L6C = 32;   //!<   obs code: E6C         (GAL)
00063 constexpr uint32_t CODE_L6X = 33;   //!<   obs code: E6B+C,LEXS+L,B3I+Q (GAL,QZS,BDS)
00064 constexpr uint32_t CODE_L6Z = 34;   //!<   obs code: E6A+B+C     (GAL)
00065 constexpr uint32_t CODE_L6S = 35;   //!<   obs code: LEXS        (QZS)
00066 constexpr uint32_t CODE_L6L = 36;   //!<   obs code: LEXL        (QZS)
00067 constexpr uint32_t CODE_L8I = 37;   //!<   obs code: E5(a+b)I    (GAL)
00068 constexpr uint32_t CODE_L8Q = 38;   //!<   obs code: E5(a+b)Q    (GAL)
00069 constexpr uint32_t CODE_L8X = 39;   //!<   obs code: E5(a+b)I+Q  (GAL)
00070 constexpr uint32_t CODE_L2I = 40;   //!<   obs code: B1I         (BDS)
00071 constexpr uint32_t CODE_L2Q = 41;   //!<   obs code: B1Q         (BDS)
00072 constexpr uint32_t CODE_L6I = 42;   //!<   obs code: B3I         (BDS)
00073 constexpr uint32_t CODE_L6Q = 43;   //!<   obs code: B3Q         (BDS)
00074 constexpr uint32_t CODE_L3I = 44;   //!<   obs code: G3I         (GLO)
00075 constexpr uint32_t CODE_L3Q = 45;   //!<   obs code: G3Q         (GLO)
00076 constexpr uint32_t CODE_L3X = 46;   //!<   obs code: G3I+Q       (GLO)
00077 constexpr uint32_t CODE_L1I = 47;   //!<   obs code: B1I         (BDS)
00078 constexpr uint32_t CODE_L1Q = 48;   //!<   obs code: B1Q         (BDS)
00079 constexpr uint32_t CODE_L5A = 49;   //!<   obs code: L5A SPS     (IRN)
00080 constexpr uint32_t CODE_L5B = 50;   //!<   obs code: L5B RS(D)   (IRN)
00081 constexpr uint32_t CODE_L5C = 51;   //!<   obs code: L5C RS(P)   (IRN)
00082 constexpr uint32_t CODE_L9A = 52;   //!<   obs code: SA SPS      (IRN)
00083 constexpr uint32_t CODE_L9B = 53;   //!<   obs code: SB RS(D)    (IRN)
00084 constexpr uint32_t CODE_L9C = 54;   //!<   obs code: SC RS(P)    (IRN)
00085 constexpr uint32_t CODE_L9X = 55;   //!<   obs code: SB+C        (IRN)
00086 constexpr uint32_t CODE_L1D = 56;   //!<   obs code: B1C data    (BDS)
00087 constexpr uint32_t CODE_L5D = 57;   //!<   obs code: B2a data    (BDS)
00088 constexpr int32_t MAXCODE = 57;     //!<   max number of obs code
00089
00090
00091 /** \} */
00092 /** \} */
00093 #endif  // GNSS_SDR_GNSS_OBS_CODES_H
```

13.744 gnss_satellite.h File Reference

Interface of the Gnss_Satellite class.

```
#include <cstdint>
#include <map>
#include <ostream>
#include <set>
#include <string>
```

Classes

- class Gnss_Satellite

  This class represents a GNSS satellite.

13.744.1 Detailed Description

Interface of the Gnss_Satellite class.

Author

Carles Fernandez-Prades, 2012. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_satellite.h.

13.745 gnss_satellite.h

Go to the documentation of this file.

```
00001 /*!
```

```
00002  * \file gnss_satellite.h
00003  * \brief  Interface of the Gnss_Satellite class
00004  * \author Carles Fernandez-Prades, 2012. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GNSS_SATELLITE_H
00019 #define GNSS_SDR_GNSS_SATELLITE_H
00020
00021 #include <cstdint>
00022 #include <map>
00023 #include <ostream>
00024 #include <set>
00025 #include <string>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 /*!
00034  * \brief This class represents a GNSS satellite.
00035  *
00036  * It contains information about the space vehicles currently operational
00037  * of GPS, Glonass, SBAS and Galileo constellations.
00038  */
00039 class Gnss_Satellite
00040 {
00041 public:
00042     Gnss_Satellite() = default;                                    //!< Default Constructor.
00043     Gnss_Satellite(const std::string& system_, uint32_t PRN_);  //!< Concrete GNSS satellite
      Constructor.
00044     ~Gnss_Satellite() = default;                                   //!< Default Destructor.
00045
00046     Gnss_Satellite(const Gnss_Satellite& other) noexcept;        //!< Copy constructor
00047     Gnss_Satellite& operator=(const Gnss_Satellite&) noexcept;   //!< Copy assignment operator
00048     Gnss_Satellite(Gnss_Satellite&& other) noexcept;             //!< Move constructor
00049     Gnss_Satellite& operator=(Gnss_Satellite&& other) noexcept;  //!< Move assignment operator
00050
00051     friend bool operator==(const Gnss_Satellite& /*sat1*/, const Gnss_Satellite& /*sat2*/);  //!<
      operator== for comparison
00052     friend std::ostream& operator<<(std::ostream& /*out*/, const Gnss_Satellite& /*sat*/);   //!<
      operator<< for pretty printing
00053
00054     void update_PRN(uint32_t PRN);                                      //!< Updates the PRN Number
      (and the corresponding block) when the slot number is decoded, only applies to GLONASS GNAV messages
00055     uint32_t get_PRN() const;                                           //!< Gets satellite's PRN
00056     int32_t get_rf_link() const;                                        //!< Gets the satellite's rf
      link
00057     std::string get_system() const;                                     //!< Gets the satellite system
      {"GPS", "Glonass", "SBAS", "Galileo", "Beidou", "QZSS"}
00058     std::string get_system_short() const;                               //!< Gets the satellite system
      {"G", "R", "SBAS", "E", "C", "J"}
00059     std::string get_block() const;                                      //!< Gets the satellite block.
      If GPS, returns {"IIA", "IIR", "IIR-M", "IIF"}
00060     std::string what_block(const std::string& system_, uint32_t PRN_);  //!< Gets the block of a given
      satellite
00061
00062 private:
00063     void set_system(const std::string& system);  // Sets the satellite system {"GPS", "Glonass",
      "SBAS", "Galileo", "Beidou", "QZSS"}.
00064     void set_PRN(uint32_t PRN);                  // Sets satellite's PRN
00065     void set_block(const std::string& system_, uint32_t PRN_);
00066     void reset();
00067     void set_rf_link(int32_t rf_link_);
00068     std::string system{};
00069     std::string block{};
00070     uint32_t PRN{};
00071     int32_t rf_link{};
00072 };
00073
00074
00075 /** \} */
00076 /** \} */
00077 #endif  // GNSS_SDR_GNSS_SATELLITE_H
```

13.746 gnss_signal.h File Reference

Implementation of the Gnss_Signal class.

```
#include "gnss_satellite.h"
#include <ostream>
#include <string>
```

Classes

- class Gnss_Signal

  This class represents a GNSS signal.

13.746.1 Detailed Description

Implementation of the Gnss_Signal class.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com Javier Arribas, 2012. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gnss_signal.h.

13.747 gnss_signal.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_signal.h
00003  * \brief  Implementation of the Gnss_Signal class
00004  * \author
00005  *  Luis Esteve, 2012. luis(at)epsilon-formacion.com
00006  *  Javier Arribas, 2012. jarribas(at)cttc.es
00007  *
00008  * -----------------------------------------------------------------------------
00009  *
00010  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00011  * This file is part of GNSS-SDR.
00012  *
00013  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00014  * SPDX-License-Identifier: GPL-3.0-or-later
00015  *
00016  * -----------------------------------------------------------------------------
00017  */
00018
00019 #ifndef GNSS_SDR_GNSS_SIGNAL_H
00020 #define GNSS_SDR_GNSS_SIGNAL_H
00021
00022 #include "gnss_satellite.h"
00023 #include <ostream>
00024 #include <string>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This class represents a GNSS signal.
00034  *
00035  * Encapsulates a specific GNSS signal (e.g., GPS L1 C/A, Galileo E1B) and its
00036  * associated satellite.
00037  */
00038 class Gnss_Signal
00039 {
00040 public:
00041     Gnss_Signal() = default;
00042     explicit Gnss_Signal(const std::string& signal_);
00043     Gnss_Signal(const Gnss_Satellite& satellite_, const std::string& signal_);
00044     ~Gnss_Signal() = default;
00045
00046     /*!
00047      * \brief Return the signal identifier string.
```

```
00048      *
00049      *  - GPS: "1C" (L1 C/A), "2S" (L2C), "L5"
00050      *  - GLONASS: "1G" (L1 C/A), "2G" (L2 C/A)
00051      *  - Galileo: "1B" (E1B), "5X" (E5a), "7X" (E5b), "6C" (E6C)
00052      *  - BeiDou: "B1" (B1I), "B3" (B3I), "1D" (B1C), "5D" (B2a data)
00053      */
00054     std::string get_signal_str() const;
00055
00056     Gnss_Satellite get_satellite() const;  //!< Get the Gnss_Satellite associated to the signal
00057
00058     friend bool operator==(const Gnss_Signal& /*sig1*/, const Gnss_Signal& /*sig2*/);    //!<
      operator== for comparison
00059     friend std::ostream& operator«(std::ostream& /*out*/, const Gnss_Signal& /*sig*/);  //!< operator«
      for pretty printing
00060
00061 private:
00062     Gnss_Satellite satellite{};
00063     std::string signal{};
00064 };
00065
00066
00067 /** \} */
00068 /** \} */
00069 #endif  // GNSS_SDR_GNSS_SIGNAL_H
```

13.748 gnss_synchro.h File Reference

Interface of the Gnss_Synchro class.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
#include <utility>
```

Classes

- class Gnss_Synchro

  This is the class that contains the information that is shared by the processing blocks.

13.748.1 Detailed Description

Interface of the Gnss_Synchro class.

Author

Luis Esteve, 2012. luis(at)epsilon-formacion.com Javier Arribas, 2012. jarribas(at)cttc.es Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gnss_synchro.h.

13.749 gnss_synchro.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gnss_synchro.h
00003  * \brief  Interface of the Gnss_Synchro class
00004  * \author
00005  *  Luis Esteve, 2012. luis(at)epsilon-formacion.com
00006  *  Javier Arribas, 2012. jarribas(at)cttc.es
00007  *  Álvaro Cebrián Juan, 2018. acebrianjuan(at)gmail.com
00008  *
00009  * -----------------------------------------------------------------------------
00010  *
00011  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00012  * This file is part of GNSS-SDR.
00013  *
00014  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00015  * SPDX-License-Identifier: GPL-3.0-or-later
00016  *
00017  * -----------------------------------------------------------------------------
00018  */
```

```
00019
00020 #ifndef GNSS_SDR_GNSS_SYNCHRO_H
00021 #define GNSS_SDR_GNSS_SYNCHRO_H
00022
00023 #include <boost/serialization/nvp.hpp>
00024 #include <cstdint>
00025 #include <utility>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters core_system_parameters
00030  * GNSS parameters
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This is the class that contains the information that is shared
00036  * by the processing blocks.
00037  */
00038 class Gnss_Synchro
00039 {
00040 public:
00041     Gnss_Synchro() = default;  //!< Default constructor
00042
00043     ~Gnss_Synchro() = default;  //!< Default destructor
00044
00045     // Satellite and signal info
00046     char System{};         //!< Set by Channel::set_signal(Gnss_Signal gnss_signal)
00047     char Signal[3]{};      //!< Set by Channel::set_signal(Gnss_Signal gnss_signal)
00048     uint32_t PRN{};        //!< Set by Channel::set_signal(Gnss_Signal gnss_signal)
00049     int32_t Channel_ID{};  //!< Set by Channel constructor
00050
00051     // Acquisition
00052     double Acq_delay_samples{};          //!< Set by Acquisition processing block
00053     double Acq_doppler_hz{};             //!< Set by Acquisition processing block
00054     uint64_t Acq_samplestamp_samples{};  //!< Set by Acquisition processing block
00055     uint32_t Acq_doppler_step{};         //!< Set by Acquisition processing block
00056
00057     // Tracking
00058     int64_t fs{};                        //!< Set by Tracking processing block
00059     double Prompt_I{};                   //!< Set by Tracking processing block
00060     double Prompt_Q{};                   //!< Set by Tracking processing block
00061     double CN0_dB_hz{};                  //!< Set by Tracking processing block
00062     double Carrier_Doppler_hz{};         //!< Set by Tracking processing block
00063     double Carrier_phase_rads{};         //!< Set by Tracking processing block
00064     double Code_phase_samples{};         //!< Set by Tracking processing block
00065     uint64_t Tracking_sample_counter{};  //!< Set by Tracking processing block
00066     int32_t correlation_length_ms{};     //!< Set by Tracking processing block
00067
00068     // Telemetry Decoder
00069     uint32_t TOW_at_current_symbol_ms{};  //!< Set by Telemetry Decoder processing block
00070
00071     // Observables
00072     double Pseudorange_m{};  //!< Set by Observables processing block
00073     double RX_time{};        //!< Set by Observables processing block
00074     double interp_TOW_ms{};  //!< Set by Observables processing block
00075
00076     // Flags
00077     bool Flag_valid_acquisition{};         //!< Set by Acquisition processing block
00078     bool Flag_valid_symbol_output{};       //!< Set by Tracking processing block
00079     bool Flag_valid_word{};                //!< Set by Telemetry Decoder processing block
00080     bool Flag_valid_pseudorange{};         //!< Set by Observables processing block
00081     bool Flag_PLL_180_deg_phase_locked{};  //!< Set by Telemetry Decoder processing block
00082     bool Flag_cycle_slip{};                //!< Set by Observables processing block
00083
00084     //! Set by Observables processing block. Signals that the carrier phase has
00085     //! jumped by half a cycle because the Costas loop phase ambiguity resolved
00086     //! by the Telemetry Decoder changed
00087     bool Flag_half_cycle_slip{};
00088
00089     //! Set by Tracking processing block. False on the first output after the
00090     //! accumulated carrier phase has been (re)initialized, that is, whenever
00091     //! the carrier phase ambiguity of the channel has changed
00092     bool Flag_carrier_phase_continuous{true};
00093
00094     /// Copy constructor
00095     Gnss_Synchro(const Gnss_Synchro& other) noexcept = default;
00096
00097     /// Copy assignment operator
00098     Gnss_Synchro& operator=(const Gnss_Synchro& rhs) noexcept
00099     {
00100         // Only do assignment if RHS is a different object from this.
00101         if (this != &rhs)
00102             {
00103                 this->System = rhs.System;
00104                 this->Signal[0] = rhs.Signal[0];
00105                 this->Signal[1] = rhs.Signal[1];
```

```
00106                 this->Signal[2] = rhs.Signal[2];
00107                 this->PRN = rhs.PRN;
00108                 this->Channel_ID = rhs.Channel_ID;
00109                 this->Acq_delay_samples = rhs.Acq_delay_samples;
00110                 this->Acq_doppler_hz = rhs.Acq_doppler_hz;
00111                 this->Acq_samplestamp_samples = rhs.Acq_samplestamp_samples;
00112                 this->Acq_doppler_step = rhs.Acq_doppler_step;
00113                 this->fs = rhs.fs;
00114                 this->Prompt_I = rhs.Prompt_I;
00115                 this->Prompt_Q = rhs.Prompt_Q;
00116                 this->CN0_dB_hz = rhs.CN0_dB_hz;
00117                 this->Carrier_Doppler_hz = rhs.Carrier_Doppler_hz;
00118                 this->Carrier_phase_rads = rhs.Carrier_phase_rads;
00119                 this->Code_phase_samples = rhs.Code_phase_samples;
00120                 this->Tracking_sample_counter = rhs.Tracking_sample_counter;
00121                 this->correlation_length_ms = rhs.correlation_length_ms;
00122                 this->TOW_at_current_symbol_ms = rhs.TOW_at_current_symbol_ms;
00123                 this->Pseudorange_m = rhs.Pseudorange_m;
00124                 this->RX_time = rhs.RX_time;
00125                 this->interp_TOW_ms = rhs.interp_TOW_ms;
00126                 this->Flag_valid_acquisition = rhs.Flag_valid_acquisition;
00127                 this->Flag_valid_symbol_output = rhs.Flag_valid_symbol_output;
00128                 this->Flag_valid_word = rhs.Flag_valid_word;
00129                 this->Flag_valid_pseudorange = rhs.Flag_valid_pseudorange;
00130                 this->Flag_PLL_180_deg_phase_locked = rhs.Flag_PLL_180_deg_phase_locked;
00131                 this->Flag_cycle_slip = rhs.Flag_cycle_slip;
00132                 this->Flag_half_cycle_slip = rhs.Flag_half_cycle_slip;
00133                 this->Flag_carrier_phase_continuous = rhs.Flag_carrier_phase_continuous;
00134             }
00135         return *this;
00136     };
00137
00138     /// Move constructor
00139     Gnss_Synchro(Gnss_Synchro&& other) noexcept = default;
00140
00141     /// Move assignment operator
00142     Gnss_Synchro& operator=(Gnss_Synchro&& other) noexcept
00143     {
00144         if (this != &other)
00145             {
00146                 this->System = other.System;
00147                 this->Signal[0] = other.Signal[0];
00148                 this->Signal[1] = other.Signal[1];
00149                 this->Signal[2] = other.Signal[2];
00150                 this->PRN = other.PRN;
00151                 this->Channel_ID = other.Channel_ID;
00152                 this->Acq_delay_samples = other.Acq_delay_samples;
00153                 this->Acq_doppler_hz = other.Acq_doppler_hz;
00154                 this->Acq_samplestamp_samples = other.Acq_samplestamp_samples;
00155                 this->Acq_doppler_step = other.Acq_doppler_step;
00156                 this->fs = other.fs;
00157                 this->Prompt_I = other.Prompt_I;
00158                 this->Prompt_Q = other.Prompt_Q;
00159                 this->CN0_dB_hz = other.CN0_dB_hz;
00160                 this->Carrier_Doppler_hz = other.Carrier_Doppler_hz;
00161                 this->Carrier_phase_rads = other.Carrier_phase_rads;
00162                 this->Code_phase_samples = other.Code_phase_samples;
00163                 this->Tracking_sample_counter = other.Tracking_sample_counter;
00164                 this->correlation_length_ms = other.correlation_length_ms;
00165                 this->TOW_at_current_symbol_ms = other.TOW_at_current_symbol_ms;
00166                 this->Pseudorange_m = other.Pseudorange_m;
00167                 this->RX_time = other.RX_time;
00168                 this->interp_TOW_ms = other.interp_TOW_ms;
00169                 this->Flag_valid_acquisition = other.Flag_valid_acquisition;
00170                 this->Flag_valid_symbol_output = other.Flag_valid_symbol_output;
00171                 this->Flag_valid_word = other.Flag_valid_word;
00172                 this->Flag_valid_pseudorange = other.Flag_valid_pseudorange;
00173                 this->Flag_PLL_180_deg_phase_locked = other.Flag_PLL_180_deg_phase_locked;
00174                 this->Flag_cycle_slip = other.Flag_cycle_slip;
00175                 this->Flag_half_cycle_slip = other.Flag_half_cycle_slip;
00176                 this->Flag_carrier_phase_continuous = other.Flag_carrier_phase_continuous;
00177
00178                 // Leave the source object in a valid but unspecified state
00179                 other.Signal[0] = '\0';
00180                 other.Signal[1] = '\0';
00181                 other.Signal[2] = '\0';
00182                 other.System = 0;
00183                 other.PRN = 0;
00184                 other.Channel_ID = 0;
00185                 other.Acq_delay_samples = 0.0;
00186                 other.Acq_doppler_hz = 0.0;
00187                 other.Acq_samplestamp_samples = 0;
00188                 other.Acq_doppler_step = 0;
00189                 other.fs = 0;
00190                 other.Prompt_I = 0.0;
00191                 other.Prompt_Q = 0.0;
00192                 other.CN0_dB_hz = 0.0;
```

```
00193                 other.Carrier_Doppler_hz = 0.0;
00194                 other.Carrier_phase_rads = 0.0;
00195                 other.Code_phase_samples = 0.0;
00196                 other.Tracking_sample_counter = 0;
00197                 other.correlation_length_ms = 0;
00198                 other.TOW_at_current_symbol_ms = 0;
00199                 other.Pseudorange_m = 0.0;
00200                 other.RX_time = 0.0;
00201                 other.interp_TOW_ms = 0.0;
00202                 other.Flag_valid_acquisition = false;
00203                 other.Flag_valid_symbol_output = false;
00204                 other.Flag_valid_word = false;
00205                 other.Flag_valid_pseudorange = false;
00206                 other.Flag_PLL_180_deg_phase_locked = false;
00207                 other.Flag_cycle_slip = false;
00208                 other.Flag_half_cycle_slip = false;
00209                 other.Flag_carrier_phase_continuous = true;
00210             }
00211         return *this;
00212     };
00213
00214     /*!
00215      * \brief This member function serializes and restores
00216      * Gnss_Synchro objects from a byte stream.
00217      */
00218     template <class Archive>
00219
00220     void serialize(Archive& ar, const unsigned int version)
00221     {
00222         if (version)
00223             {
00224             };
00225         // Satellite and signal info
00226         ar& BOOST_SERIALIZATION_NVP(System);
00227         ar& BOOST_SERIALIZATION_NVP(Signal);
00228         ar& BOOST_SERIALIZATION_NVP(PRN);
00229         ar& BOOST_SERIALIZATION_NVP(Channel_ID);
00230         // Acquisition
00231         ar& BOOST_SERIALIZATION_NVP(Acq_delay_samples);
00232         ar& BOOST_SERIALIZATION_NVP(Acq_doppler_hz);
00233         ar& BOOST_SERIALIZATION_NVP(Acq_samplestamp_samples);
00234         ar& BOOST_SERIALIZATION_NVP(Acq_doppler_step);
00235         // Tracking
00236         ar& BOOST_SERIALIZATION_NVP(fs);
00237         ar& BOOST_SERIALIZATION_NVP(Prompt_I);
00238         ar& BOOST_SERIALIZATION_NVP(Prompt_Q);
00239         ar& BOOST_SERIALIZATION_NVP(CN0_dB_hz);
00240         ar& BOOST_SERIALIZATION_NVP(Carrier_Doppler_hz);
00241         ar& BOOST_SERIALIZATION_NVP(Carrier_phase_rads);
00242         ar& BOOST_SERIALIZATION_NVP(Code_phase_samples);
00243         ar& BOOST_SERIALIZATION_NVP(Tracking_sample_counter);
00244         ar& BOOST_SERIALIZATION_NVP(correlation_length_ms);
00245         // Telemetry Decoder
00246         ar& BOOST_SERIALIZATION_NVP(TOW_at_current_symbol_ms);
00247         // Observables
00248         ar& BOOST_SERIALIZATION_NVP(Pseudorange_m);
00249         ar& BOOST_SERIALIZATION_NVP(RX_time);
00250         ar& BOOST_SERIALIZATION_NVP(interp_TOW_ms);
00251         // Flags
00252         ar& BOOST_SERIALIZATION_NVP(Flag_valid_acquisition);
00253         ar& BOOST_SERIALIZATION_NVP(Flag_valid_symbol_output);
00254         ar& BOOST_SERIALIZATION_NVP(Flag_valid_word);
00255         ar& BOOST_SERIALIZATION_NVP(Flag_valid_pseudorange);
00256         ar& BOOST_SERIALIZATION_NVP(Flag_PLL_180_deg_phase_locked);
00257         ar& BOOST_SERIALIZATION_NVP(Flag_cycle_slip);
00258         ar& BOOST_SERIALIZATION_NVP(Flag_half_cycle_slip);
00259         ar& BOOST_SERIALIZATION_NVP(Flag_carrier_phase_continuous);
00260     }
00261 };
00262
00263
00264 /** \} */
00265 /** \} */
00266 #endif  // GNSS_SDR_GNSS_SYNCHRO_H
```

13.750 gps_acq_assist.h File Reference

Interface of a GPS RRLL ACQUISITION ASSISTACE storage.

`#include <cstdint>`

Classes

- class Gps_Acq_Assist

  This class is a storage for the GPS GSM RRLL acquisition assistance data as described in Digital cellular telecommunications system (Phase 2+); Location Services (LCS); Mobile Station (MS) - Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP) (3GPP TS 44.031 version 5.12.0 Release 5).

13.750.1 Detailed Description

Interface of a GPS RRLL ACQUISITION ASSISTACE storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_acq_assist.h.

13.751 gps_acq_assist.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_acq_assist.h
00003  * \brief  Interface of a GPS RRLL ACQUISITION ASSISTACE storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_ACQ_ASSIST_H
00019 #define GNSS_SDR_GPS_ACQ_ASSIST_H
00020
00021 #include <cstdint>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief This class is a storage for the GPS GSM RRLL acquisition assistance data as described in
00031  * Digital cellular telecommunications system (Phase 2+);
00032  * Location Services (LCS);
00033  * Mobile Station (MS) - Serving Mobile Location Centre (SMLC)
00034  * Radio Resource LCS Protocol (RRLP)
00035  * (3GPP TS 44.031 version 5.12.0 Release 5)
00036  */
00037 class Gps_Acq_Assist
00038 {
00039 public:
00040     /*!
00041      * Default constructor
00042      */
00043     Gps_Acq_Assist() = default;
00044
00045     uint32_t PRN{};               //!< SV PRN NUMBER
00046     double tow{};                 //!< Time Of Week assigned to the acquisition data
00047     double Doppler0{};            //!< Doppler (0 order term) [Hz]
00048     double Doppler1{};            //!< Doppler (1 order term) [Hz]
00049     double dopplerUncertainty{};  //!< Doppler Uncertainty [Hz]
00050     double Code_Phase{};          //!< Code phase [chips]
00051     double Code_Phase_int{};      //!< Integer Code Phase [1 C/A code period]
00052     double GPS_Bit_Number{};      //!< GPS Bit Number
00053     double Code_Phase_window{};   //!< Code Phase search window [chips]
00054     double Azimuth{};             //!< Satellite Azimuth [deg]
00055     double Elevation{};           //!< Satellite Elevation [deg]
00056 };
```

```
00057
00058
00059 /** \} */
00060 /** \} */
00061 #endif  // GNSS_SDR_GPS_ACQ_ASSIST_H
```

13.752 gps_almanac.h File Reference

Interface of a GPS ALMANAC storage.

```
#include "gnss_almanac.h"
#include <boost/serialization/nvp.hpp>
```

Classes

- class Gps_Almanac

  This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M.

13.752.1 Detailed Description

Interface of a GPS ALMANAC storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_almanac.h.

13.753 gps_almanac.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_almanac.h
00003  * \brief  Interface of a GPS ALMANAC storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_ALMANAC_H
00019 #define GNSS_SDR_GPS_ALMANAC_H
00020
00021 #include "gnss_almanac.h"
00022 #include <boost/serialization/nvp.hpp>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class is a storage for the GPS SV ALMANAC data as described in IS-GPS-200M
00032  *
00033  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
00034  */
00035 class Gps_Almanac : public Gnss_Almanac
00036 {
00037 public:
00038     /*!
00039      * Default constructor
00040      */
```

```
00041     Gps_Almanac()
00042     {
00043         this->System = 'G';
00044     };
00045
00046     void set_system(char system)
00047     {
00048         this->System = system;
00049     }
00050
00051     char get_system() const
00052     {
00053         return this->System;
00054     }
00055
00056     int32_t SV_health{};  //!< SV Health
00057     int32_t AS_status{};  //!< Anti-Spoofing Flags and SV Configuration
00058
00059     template <class Archive>
00060
00061     void serialize(Archive& ar, const unsigned int version)
00062     {
00063         if (version)
00064             {
00065             };
00066         ar& BOOST_SERIALIZATION_NVP(PRN);
00067         ar& BOOST_SERIALIZATION_NVP(delta_i);
00068         ar& BOOST_SERIALIZATION_NVP(toa);
00069         ar& BOOST_SERIALIZATION_NVP(WNa);
00070         ar& BOOST_SERIALIZATION_NVP(M_0);
00071         ar& BOOST_SERIALIZATION_NVP(ecc);
00072         ar& BOOST_SERIALIZATION_NVP(sqrtA);
00073         ar& BOOST_SERIALIZATION_NVP(OMEGA_0);
00074         ar& BOOST_SERIALIZATION_NVP(omega);
00075         ar& BOOST_SERIALIZATION_NVP(OMEGAdot);
00076         ar& BOOST_SERIALIZATION_NVP(af0);
00077         ar& BOOST_SERIALIZATION_NVP(af1);
00078         ar& BOOST_SERIALIZATION_NVP(SV_health);
00079         ar& BOOST_SERIALIZATION_NVP(AS_status);
00080     }
00081 };
00082
00083
00084 /** \} */
00085 /** \} */
00086 #endif  // GNSS_SDR_GPS_ALMANAC_H
```

13.754 GPS_CNAV.h File Reference

Defines parameters for GPS CNAV.

```
#include "MATH_CONSTANTS.h"
#include <cstdint>
#include <utility>
#include <vector>
```

Functions

- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PRN** ({{9, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_MSG_TYPE** ({{15, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOW** ({{21, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALERT_FLAG** ({{38, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN** ({{39, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_HEALTH** ({{52, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOP1** ({{55, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA** ({{66, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOE1** ({{71, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_A** ({{82, 26}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A_DOT** ({{108, 25}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_N0** ({{133, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_N0_DOT** ({{150, 23}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_M0** ({{173, 33}})

- const std::vector< std::pair< int32_t, int32_t > > **CNAV_E_ECCENTRICITY** ({{206, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_OMEGA** ({{239, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_INTEGRITY_FLAG** ({{272, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_L2_PHASING_FLAG** ({{273, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOE2** ({{39, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_OMEGA0** ({{50, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_I0** ({{83, 33}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_OMEGA_DOT** ({{116, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_I0_DOT** ({{133, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CIS** ({{148, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CIC** ({{164, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CRS** ({{180, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CRC** ({{204, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CUS** ({{228, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_CUC** ({{249, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOP2** ({{39, 11}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED0** ({{50, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED1** ({{55, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_URA_NED2** ({{58, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOC** ({{61, 11}})
- const std::vector< std::pair< int, int > > **CNAV_AF0** ({{72, 26}})
- const std::vector< std::pair< int, int > > **CNAV_AF1** ({{98, 20}})
- const std::vector< std::pair< int, int > > **CNAV_AF2** ({{118, 10}})
- const std::vector< std::pair< int, int > > **CNAV_TGD** ({{128, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL1** ({{141, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL2** ({{154, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL5I** ({{167, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ISCL5Q** ({{180, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA0** ({{193, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA1** ({{201, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA2** ({{209, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_ALPHA3** ({{217, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA0** ({{225, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA1** ({{233, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA2** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_BETA3** ({{249, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WNOP** ({{257, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_T_EOP** ({{128, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_X** ({{144, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_X_DOT** ({{165, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_Y** ({{180, 21}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_PM_Y_DOT** ({{201, 15}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_UT1_GPS** ({{216, 31}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_UT1_GPS_DOT** ({{247, 19}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A0** ({{128, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A1** ({{144, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_A2** ({{157, 7}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_TLS** ({{164, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_TOT** ({{172, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN_OT** ({{188, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_WN_LSF** ({{201, 13}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DN** ({{214, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **CNAV_DELTA_TLSF** ({{218, 8}})

Variables

- constexpr int32_t GPS_CNAV_DATA_PAGE_BITS = 300
- constexpr int32_t CNAV_TOW_LSB = 6
- constexpr int32_t CNAV_TOP1_LSB = 300
- constexpr int32_t CNAV_TOE1_LSB = 300
- constexpr double CNAV_DELTA_A_LSB = TWO_N9
- constexpr double CNAV_A_DOT_LSB = TWO_N21
- constexpr double CNAV_DELTA_N0_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI
- constexpr double CNAV_M0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_E_ECCENTRICITY_LSB = TWO_N34
- constexpr double CNAV_OMEGA_LSB = TWO_N32 * GNSS_PI
- constexpr int32_t CNAV_TOE2_LSB = 300
- constexpr double CNAV_OMEGA0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_I0_LSB = TWO_N32 * GNSS_PI
- constexpr double CNAV_DELTA_OMEGA_DOT_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_I0_DOT_LSB = TWO_N44 * GNSS_PI
- constexpr double CNAV_CIS_LSB = TWO_N30
- constexpr double CNAV_CIC_LSB = TWO_N30
- constexpr double CNAV_CRS_LSB = TWO_N8
- constexpr double CNAV_CRC_LSB = TWO_N8
- constexpr double CNAV_CUS_LSB = TWO_N30
- constexpr double CNAV_CUC_LSB = TWO_N30
- constexpr int32_t CNAV_TOP2_LSB = 300
- constexpr int32_t CNAV_TOC_LSB = 300
- constexpr double CNAV_AF0_LSB = TWO_N35
- constexpr double CNAV_AF1_LSB = TWO_N48
- constexpr double CNAV_AF2_LSB = TWO_N60
- constexpr double CNAV_TGD_LSB = TWO_N35
- constexpr double CNAV_ISCL1_LSB = TWO_N35
- constexpr double CNAV_ISCL2_LSB = TWO_N35
- constexpr double CNAV_ISCL5I_LSB = TWO_N35
- constexpr double CNAV_ISCL5Q_LSB = TWO_N35
- constexpr double CNAV_ALPHA0_LSB = TWO_N30
- constexpr double CNAV_ALPHA1_LSB = TWO_N27
- constexpr double CNAV_ALPHA2_LSB = TWO_N24
- constexpr double CNAV_ALPHA3_LSB = TWO_N24
- constexpr double CNAV_BETA0_LSB = TWO_P11
- constexpr double CNAV_BETA1_LSB = TWO_P14
- constexpr double CNAV_BETA2_LSB = TWO_P16
- constexpr double CNAV_BETA3_LSB = TWO_P16
- constexpr int32_t CNAV_T_EOP_LSB = TWO_P4
- constexpr double CNAV_PM_X_LSB = TWO_N20
- constexpr double CNAV_PM_X_DOT_LSB = TWO_N21
- constexpr double CNAV_PM_Y_LSB = TWO_N20
- constexpr double CNAV_PM_Y_DOT_LSB = TWO_N21
- constexpr double CNAV_DELTA_UT1_GPS_LSB = TWO_N23
- constexpr double CNAV_DELTA_UT1_GPS_DOT_LSB = TWO_N25
- constexpr double CNAV_A0_LSB = TWO_N35
- constexpr double CNAV_A1_LSB = TWO_N51
- constexpr double CNAV_A2_LSB = TWO_N68
- constexpr int32_t CNAV_DELTA_TLS_LSB = 1
- constexpr int32_t CNAV_TOT_LSB = TWO_P4
- constexpr int32_t CNAV_WN_OT_LSB = 1

- constexpr int32_t CNAV_WN_LSF_LSB = 1
- constexpr int32_t CNAV_DN_LSB = 1
- constexpr int32_t CNAV_DELTA_TLSF_LSB = 1
- constexpr double CNAV_A_REF = 26559710.0
- constexpr double CNAV_OMEGA_DOT_REF = -2.6e-9
- constexpr double CNAV_QZSS_A_REF = 42164200.0

13.754.1 Detailed Description

Defines parameters for GPS CNAV.

Author

Antonio Ramos, 2017. antonio.ramos(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file GPS_CNAV.h.

13.755 GPS_CNAV.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file GPS_CNAV.h
00003  * \brief  Defines parameters for GPS CNAV
00004  * \author Antonio Ramos, 2017. antonio.ramos(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_CNAV_H
00019 #define GNSS_SDR_GPS_CNAV_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include <cstdint>
00023 #include <utility>  // std::pair
00024 #include <vector>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 // CNAV GPS NAVIGATION MESSAGE STRUCTURE
00033 // NAVIGATION MESSAGE FIELDS POSITIONS (from IS-GPS-200M Appendix III)
00034
00035 constexpr int32_t GPS_CNAV_DATA_PAGE_BITS = 300;
00036
00037 // common to all messages
00038 const std::vector<std::pair<int32_t, int32_t> > CNAV_PRN({{9, 6}});
00039 const std::vector<std::pair<int32_t, int32_t> > CNAV_MSG_TYPE({{15, 6}});
00040 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOW({{21, 17}});  // GPS Time Of Week in seconds
00041 constexpr int32_t CNAV_TOW_LSB = 6;
00042 const std::vector<std::pair<int32_t, int32_t> > CNAV_ALERT_FLAG({{38, 1}});
00043
00044 // MESSAGE TYPE 10 (Ephemeris 1)
00045 const std::vector<std::pair<int32_t, int32_t> > CNAV_WN({{39, 13}});
00046 const std::vector<std::pair<int32_t, int32_t> > CNAV_HEALTH({{52, 3}});
00047 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOP1({{55, 11}});
00048 constexpr int32_t CNAV_TOP1_LSB = 300;
00049 const std::vector<std::pair<int32_t, int32_t> > CNAV_URA({{66, 5}});
00050
00051 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOE1({{71, 11}});
00052 constexpr int32_t CNAV_TOE1_LSB = 300;
00053
00054 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_A({{82, 26}});  // Relative to AREF =
      26,559,710 meters
```

```
00055 constexpr double CNAV_DELTA_A_LSB = TWO_N9;
00056
00057 const std::vector<std::pair<int32_t, int32_t> > CNAV_A_DOT({{108, 25}});
00058 constexpr double CNAV_A_DOT_LSB = TWO_N21;
00059
00060 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_N0({{133, 17}});
00061 constexpr double CNAV_DELTA_N0_LSB = TWO_N44 * GNSS_PI;  // semi-circles to radians
00062 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_N0_DOT({{150, 23}});
00063 constexpr double CNAV_DELTA_N0_DOT_LSB = TWO_N57 * GNSS_PI;  // semi-circles to radians
00064 const std::vector<std::pair<int32_t, int32_t> > CNAV_M0({{173, 33}});
00065 constexpr double CNAV_M0_LSB = TWO_N32 * GNSS_PI;  // semi-circles to radians
00066 const std::vector<std::pair<int32_t, int32_t> > CNAV_E_ECCENTRICITY({{206, 33}});
00067 constexpr double CNAV_E_ECCENTRICITY_LSB = TWO_N34;
00068 const std::vector<std::pair<int32_t, int32_t> > CNAV_OMEGA({{239, 33}});
00069 constexpr double CNAV_OMEGA_LSB = TWO_N32 * GNSS_PI;  // semi-circles to radians
00070 const std::vector<std::pair<int32_t, int32_t> > CNAV_INTEGRITY_FLAG({{272, 1}});
00071 const std::vector<std::pair<int32_t, int32_t> > CNAV_L2_PHASING_FLAG({{273, 1}});
00072
00073 // MESSAGE TYPE 11 (Ephemeris 2)
00074 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOE2({{39, 11}});
00075 constexpr int32_t CNAV_TOE2_LSB = 300;
00076 const std::vector<std::pair<int32_t, int32_t> > CNAV_OMEGA0({{50, 33}});
00077 constexpr double CNAV_OMEGA0_LSB = TWO_N32 * GNSS_PI;  // semi-circles to radians
00078 const std::vector<std::pair<int32_t, int32_t> > CNAV_I0({{83, 33}});
00079 constexpr double CNAV_I0_LSB = TWO_N32 * GNSS_PI;                                  // semi-circles to
     radians
00080 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_OMEGA_DOT({{116, 17}});  // Relative to REF
     = -2.6 x 10-9 semi-circles/second.
00081 constexpr double CNAV_DELTA_OMEGA_DOT_LSB = TWO_N44 * GNSS_PI;                     // semi-circles to
     radians
00082 const std::vector<std::pair<int32_t, int32_t> > CNAV_I0_DOT({{133, 15}});
00083 constexpr double CNAV_I0_DOT_LSB = TWO_N44 * GNSS_PI;  // semi-circles to radians
00084 const std::vector<std::pair<int32_t, int32_t> > CNAV_CIS({{148, 16}});
00085 constexpr double CNAV_CIS_LSB = TWO_N30;
00086 const std::vector<std::pair<int32_t, int32_t> > CNAV_CIC({{164, 16}});
00087 constexpr double CNAV_CIC_LSB = TWO_N30;
00088 const std::vector<std::pair<int32_t, int32_t> > CNAV_CRS({{180, 24}});
00089 constexpr double CNAV_CRS_LSB = TWO_N8;
00090 const std::vector<std::pair<int32_t, int32_t> > CNAV_CRC({{204, 24}});
00091 constexpr double CNAV_CRC_LSB = TWO_N8;
00092 const std::vector<std::pair<int32_t, int32_t> > CNAV_CUS({{228, 21}});
00093 constexpr double CNAV_CUS_LSB = TWO_N30;
00094 const std::vector<std::pair<int32_t, int32_t> > CNAV_CUC({{249, 21}});
00095 constexpr double CNAV_CUC_LSB = TWO_N30;
00096
00097
00098 // MESSAGE TYPE 30 (CLOCK, IONO, GROUP DELAY)
00099 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOP2({{39, 11}});
00100 constexpr int32_t CNAV_TOP2_LSB = 300;
00101 const std::vector<std::pair<int32_t, int32_t> > CNAV_URA_NED0({{50, 5}});
00102 const std::vector<std::pair<int32_t, int32_t> > CNAV_URA_NED1({{55, 3}});
00103 const std::vector<std::pair<int32_t, int32_t> > CNAV_URA_NED2({{58, 3}});
00104 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOC({{61, 11}});
00105 constexpr int32_t CNAV_TOC_LSB = 300;
00106 const std::vector<std::pair<int, int> > CNAV_AF0({{72, 26}});
00107 constexpr double CNAV_AF0_LSB = TWO_N35;
00108 const std::vector<std::pair<int, int> > CNAV_AF1({{98, 20}});
00109 constexpr double CNAV_AF1_LSB = TWO_N48;
00110 const std::vector<std::pair<int, int> > CNAV_AF2({{118, 10}});
00111 constexpr double CNAV_AF2_LSB = TWO_N60;
00112 const std::vector<std::pair<int, int> > CNAV_TGD({{128, 13}});
00113 constexpr double CNAV_TGD_LSB = TWO_N35;
00114 const std::vector<std::pair<int32_t, int32_t> > CNAV_ISCL1({{141, 13}});
00115 constexpr double CNAV_ISCL1_LSB = TWO_N35;
00116 const std::vector<std::pair<int32_t, int32_t> > CNAV_ISCL2({{154, 13}});
00117 constexpr double CNAV_ISCL2_LSB = TWO_N35;
00118 const std::vector<std::pair<int32_t, int32_t> > CNAV_ISCL5I({{167, 13}});
00119 constexpr double CNAV_ISCL5I_LSB = TWO_N35;
00120 const std::vector<std::pair<int32_t, int32_t> > CNAV_ISCL5Q({{180, 13}});
00121 constexpr double CNAV_ISCL5Q_LSB = TWO_N35;
00122 // Ionospheric parameters
00123 const std::vector<std::pair<int32_t, int32_t> > CNAV_ALPHA0({{193, 8}});
00124 constexpr double CNAV_ALPHA0_LSB = TWO_N30;
00125 const std::vector<std::pair<int32_t, int32_t> > CNAV_ALPHA1({{201, 8}});
00126 constexpr double CNAV_ALPHA1_LSB = TWO_N27;
00127 const std::vector<std::pair<int32_t, int32_t> > CNAV_ALPHA2({{209, 8}});
00128 constexpr double CNAV_ALPHA2_LSB = TWO_N24;
00129 const std::vector<std::pair<int32_t, int32_t> > CNAV_ALPHA3({{217, 8}});
00130 constexpr double CNAV_ALPHA3_LSB = TWO_N24;
00131 const std::vector<std::pair<int32_t, int32_t> > CNAV_BETA0({{225, 8}});
00132 constexpr double CNAV_BETA0_LSB = TWO_P11;
00133 const std::vector<std::pair<int32_t, int32_t> > CNAV_BETA1({{233, 8}});
00134 constexpr double CNAV_BETA1_LSB = TWO_P14;
00135 const std::vector<std::pair<int32_t, int32_t> > CNAV_BETA2({{241, 8}});
00136 constexpr double CNAV_BETA2_LSB = TWO_P16;
00137 const std::vector<std::pair<int32_t, int32_t> > CNAV_BETA3({{249, 8}});
00138 constexpr double CNAV_BETA3_LSB = TWO_P16;
```

```
00139 const std::vector<std::pair<int32_t, int32_t> > CNAV_WNOP({{257, 8}});
00140
00141
00142 // MESSAGE TYPE 32 (CLOCK and EARTH ORIENTATION PARAMETERS)
00143 const std::vector<std::pair<int32_t, int32_t> > CNAV_T_EOP({{128, 16}});
00144 constexpr int32_t CNAV_T_EOP_LSB = TWO_P4;
00145 const std::vector<std::pair<int32_t, int32_t> > CNAV_PM_X({{144, 21}});
00146 constexpr double CNAV_PM_X_LSB = TWO_N20;
00147 const std::vector<std::pair<int32_t, int32_t> > CNAV_PM_X_DOT({{165, 15}});
00148 constexpr double CNAV_PM_X_DOT_LSB = TWO_N21;
00149 const std::vector<std::pair<int32_t, int32_t> > CNAV_PM_Y({{180, 21}});
00150 constexpr double CNAV_PM_Y_LSB = TWO_N20;
00151 const std::vector<std::pair<int32_t, int32_t> > CNAV_PM_Y_DOT({{201, 15}});
00152 constexpr double CNAV_PM_Y_DOT_LSB = TWO_N21;
00153 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_UT1_GPS({{216, 31}});
00154 constexpr double CNAV_DELTA_UT1_GPS_LSB = TWO_N23;
00155 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_UT1_GPS_DOT({{247, 19}});
00156 constexpr double CNAV_DELTA_UT1_GPS_DOT_LSB = TWO_N25;
00157
00158
00159 // MESSAGE TYPE 33 (CLOCK and UTC)
00160 const std::vector<std::pair<int32_t, int32_t> > CNAV_A0({{128, 16}});
00161 constexpr double CNAV_A0_LSB = TWO_N35;
00162 const std::vector<std::pair<int32_t, int32_t> > CNAV_A1({{144, 13}});
00163 constexpr double CNAV_A1_LSB = TWO_N51;
00164 const std::vector<std::pair<int32_t, int32_t> > CNAV_A2({{157, 7}});
00165 constexpr double CNAV_A2_LSB = TWO_N68;
00166 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_TLS({{164, 8}});
00167 constexpr int32_t CNAV_DELTA_TLS_LSB = 1;
00168 const std::vector<std::pair<int32_t, int32_t> > CNAV_TOT({{172, 16}});
00169 constexpr int32_t CNAV_TOT_LSB = TWO_P4;
00170 const std::vector<std::pair<int32_t, int32_t> > CNAV_WN_OT({{188, 13}});
00171 constexpr int32_t CNAV_WN_OT_LSB = 1;
00172 const std::vector<std::pair<int32_t, int32_t> > CNAV_WN_LSF({{201, 13}});
00173 constexpr int32_t CNAV_WN_LSF_LSB = 1;
00174 const std::vector<std::pair<int32_t, int32_t> > CNAV_DN({{214, 4}});
00175 constexpr int32_t CNAV_DN_LSB = 1;
00176 const std::vector<std::pair<int32_t, int32_t> > CNAV_DELTA_TLSF({{218, 8}});
00177 constexpr int32_t CNAV_DELTA_TLSF_LSB = 1;
00178
00179 constexpr double CNAV_A_REF = 26559710.0;       // [m] See IS-GPS-200M, Table 30-I.
00180 constexpr double CNAV_OMEGA_DOT_REF = -2.6e-9;  // [semicircles / s], see IS-GPS-200M, Table 30-I.
00181 constexpr double CNAV_QZSS_A_REF = 42164200.0;
00182
00183 // TODO: Add more frames (Almanac, etc...)
00184
00185
00186 /** \} */
00187 /** \} */
00188 #endif  // GNSS_SDR_GPS_CNAV_H
```

13.756 gps_cnav_eop.h File Reference

Storage for GPS/QZSS CNAV Earth orientation parameters.

`#include <cstdint>`

Classes

- class Gps_CNAV_Eop

  Storage for the Earth orientation parameters broadcast in GPS/QZSS CNAV Message Type 32, as described in IS-GPS-200M.

13.756.1 Detailed Description

Storage for GPS/QZSS CNAV Earth orientation parameters.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_cnav_eop.h.

13.757 gps_cnav_eop.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_cnav_eop.h
00003  * \brief Storage for GPS/QZSS CNAV Earth orientation parameters
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_CNAV_EOP_H
00019 #define GNSS_SDR_GPS_CNAV_EOP_H
00020
00021
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief Storage for the Earth orientation parameters broadcast in GPS/QZSS
00032  * CNAV Message Type 32, as described in IS-GPS-200M.
00033  */
00034 class Gps_CNAV_Eop
00035 {
00036 public:
00037     uint32_t PRN{};              //!< Transmitting satellite PRN
00038     double tow{};                //!< Message transmission time [s of GPS week]
00039     double t_eop{};              //!< EOP reference time [s of GPS week]
00040     double pm_x{};               //!< X-axis polar motion [arc-seconds]
00041     double pm_x_dot{};           //!< X-axis polar motion rate [arc-seconds/day]
00042     double pm_y{};               //!< Y-axis polar motion [arc-seconds]
00043     double pm_y_dot{};           //!< Y-axis polar motion rate [arc-seconds/day]
00044     double delta_ut1_gps{};      //!< UT1-GPS difference [s]
00045     double delta_ut1_gps_dot{};  //!< UT1-GPS difference rate [s/day]
00046     bool valid{};
00047 };
00048
00049
00050 /** \} */
00051 /** \} */
00052 #endif  // GNSS_SDR_GPS_CNAV_EOP_H
```

13.758 gps_cnav_ephemeris.h File Reference

Interface of a GPS CNAV EPHEMERIS storage.

```
#include "gnss_ephemeris.h"
#include <boost/serialization/nvp.hpp>
#include <boost/serialization/version.hpp>
#include <cstdint>
```

Classes

- class Gps_CNAV_Ephemeris

  This is a storage class for the GPS CNAV ephemeris data as described in IS-GPS-200M.

13.758.1 Detailed Description

Interface of a GPS CNAV EPHEMERIS storage.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_cnav_ephemeris.h.

13.759 gps_cnav_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_cnav_ephemeris.h
00003  * \brief  Interface of a GPS CNAV EPHEMERIS storage
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_CNAV_EPHEMERIS_H
00019 #define GNSS_SDR_GPS_CNAV_EPHEMERIS_H
00020
00021 #include "gnss_ephemeris.h"
00022 #include <boost/serialization/nvp.hpp>
00023 #include <boost/serialization/version.hpp>
00024 #include <cstdint>
00025
00026 /** \addtogroup Core
00027  * \{ */
00028 /** \addtogroup System_Parameters
00029  * \{ */
00030
00031
00032 /*!
00033  * \brief This is a storage class for the GPS CNAV ephemeris data as described
00034  * in IS-GPS-200M
00035  *
00036  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
00037  */
00038 class Gps_CNAV_Ephemeris : public Gnss_Ephemeris
00039 {
00040 public:
00041     /*!
00042      * Constructor
00043      */
00044     Gps_CNAV_Ephemeris()
00045     {
00046         this->System = 'G';
00047     }
00048
00049     double delta_A{};         //!< Semi-major axis difference at reference time
00050     double delta_OMEGAdot{};  //!< Rate of Right Ascension  difference [semi-circles/s]
00051     int32_t toe1{};           //!< Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M)
    [s]
00052     int32_t toe2{};           //!< Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M)
    [s]
00053     int32_t signal_health{};  //!< Signal health (L1/L2/L5)
00054     int32_t WNop{-1};         //!< CEI data sequence propagation week number from MT30, modulo 256
00055     int32_t top{};            //!< Data predict time of week
00056     int32_t URAED{};          //!< ED Accuracy Index from MT10
00057
00058     int32_t URANED0{};  //!< NED Accuracy Index from MT30-37
00059     int32_t URANED1{};  //!< NED Accuracy Change Index from MT30-37
00060     int32_t URANED2{};  //!< NED Accuracy Change Rate Index from MT30-37
00061
00062     int32_t URA{};  //!< Legacy alias for URAED
00063     double URA0{};  //!< Legacy alias for URANED0
00064     double URA1{};  //!< Legacy alias for URANED1
00065     double URA2{};  //!< Legacy alias for URANED2
00066
00067     // Group Delay Differential Parameters
00068     double TGD{};  //!< Estimated Group Delay Differential: L1-L2 correction term only for the benefit
    of "L1 P(Y)" or "L2 P(Y)" s users [s]
00069     double ISCL1{};
```

```
00070     double ISCL2{};
00071     double ISCL5I{};
00072     double ISCL5Q{};
00073
00074     /*! \brief If true, enhanced level of integrity assurance.
00075      *
00076      *  If false, indicates that the conveying signal is provided with the
00077      *  legacy level of integrity assurance. That is, the probability that the
00078      *  instantaneous URE of the conveying signal exceeds 4.42 times the upper
00079      *  bound value of the current broadcast URA index, for more than 5.2
00080      *  seconds, without an accompanying alert, is less than 1E-5 per hour. If
00081      *  true, indicates that the conveying signal is provided with an enhanced
00082      *  level of integrity assurance. That is, the probability that the
00083      *  instantaneous URE of the conveying signal exceeds 5.73 times the upper
00084      *  bound value of the current broadcast URA index, for more than 5.2
00085      *  seconds, without an accompanying alert, is less than 1E-8 per hour.
00086      */
00087     bool integrity_status_flag{};
00088     bool l2c_phasing_flag{};
00089     bool alert_flag{};          //!< If true, indicates that the SV URA may be worse than indicated in
      d_SV_accuracy, use that SV at our own risk.
00090     bool antispoofing_flag{};  //!< If true, the AntiSpoofing mode is ON in that SV
00091
00092     template <class Archive>
00093
00094     /*!
00095      * \brief Serialize is a boost standard method to be called by the boost XML
00096      * serialization. Here is used to save the ephemeris data on disk file.
00097      */
00098     inline void serialize(Archive& archive, const uint32_t version)
00099     {
00100         archive& BOOST_SERIALIZATION_NVP(PRN);
00101         archive& BOOST_SERIALIZATION_NVP(M_0);
00102         archive& BOOST_SERIALIZATION_NVP(delta_n);
00103         archive& BOOST_SERIALIZATION_NVP(ecc);
00104         archive& BOOST_SERIALIZATION_NVP(sqrtA);
00105         archive& BOOST_SERIALIZATION_NVP(OMEGA_0);
00106         archive& BOOST_SERIALIZATION_NVP(i_0);
00107         archive& BOOST_SERIALIZATION_NVP(omega);
00108         archive& BOOST_SERIALIZATION_NVP(OMEGAdot);
00109         archive& BOOST_SERIALIZATION_NVP(idot);
00110         archive& BOOST_SERIALIZATION_NVP(Cuc);
00111         archive& BOOST_SERIALIZATION_NVP(Cus);
00112         archive& BOOST_SERIALIZATION_NVP(Crc);
00113         archive& BOOST_SERIALIZATION_NVP(Crs);
00114         archive& BOOST_SERIALIZATION_NVP(Cic);
00115         archive& BOOST_SERIALIZATION_NVP(Cis);
00116         archive& BOOST_SERIALIZATION_NVP(toe);
00117         archive& BOOST_SERIALIZATION_NVP(toc);
00118         archive& BOOST_SERIALIZATION_NVP(af0);
00119         archive& BOOST_SERIALIZATION_NVP(af1);
00120         archive& BOOST_SERIALIZATION_NVP(af2);
00121         archive& BOOST_SERIALIZATION_NVP(WN);
00122         archive& BOOST_SERIALIZATION_NVP(tow);
00123         archive& BOOST_SERIALIZATION_NVP(satClkDrift);
00124         archive& BOOST_SERIALIZATION_NVP(dtr);
00125
00126         archive& BOOST_SERIALIZATION_NVP(toe1);
00127         archive& BOOST_SERIALIZATION_NVP(toe2);
00128         if (version > 0)
00129             {
00130                 archive& BOOST_SERIALIZATION_NVP(WNop);
00131                 archive& BOOST_SERIALIZATION_NVP(top);
00132                 archive& BOOST_SERIALIZATION_NVP(URAED);
00133                 archive& BOOST_SERIALIZATION_NVP(URANED0);
00134                 archive& BOOST_SERIALIZATION_NVP(URANED1);
00135                 archive& BOOST_SERIALIZATION_NVP(URANED2);
00136                 archive& BOOST_SERIALIZATION_NVP(URA);
00137                 archive& BOOST_SERIALIZATION_NVP(URA0);
00138                 archive& BOOST_SERIALIZATION_NVP(URA1);
00139                 archive& BOOST_SERIALIZATION_NVP(URA2);
00140             }
00141         if (version > 1)
00142             {
00143                 archive& BOOST_SERIALIZATION_NVP(delta_ndot);
00144             }
00145         archive& BOOST_SERIALIZATION_NVP(TGD);
00146         archive& BOOST_SERIALIZATION_NVP(ISCL1);
00147         archive& BOOST_SERIALIZATION_NVP(ISCL2);
00148         archive& BOOST_SERIALIZATION_NVP(ISCL5I);
00149         archive& BOOST_SERIALIZATION_NVP(ISCL5Q);
00150         archive& BOOST_SERIALIZATION_NVP(delta_A);
00151         archive& BOOST_SERIALIZATION_NVP(Adot);
00152         archive& BOOST_SERIALIZATION_NVP(delta_OMEGAdot);
00153         archive& BOOST_SERIALIZATION_NVP(integrity_status_flag);
00154         archive& BOOST_SERIALIZATION_NVP(l2c_phasing_flag);
00155         archive& BOOST_SERIALIZATION_NVP(alert_flag);
```

```
00156         archive& BOOST_SERIALIZATION_NVP(antispoofing_flag);
00157     }
00158 };
00159
00160 BOOST_CLASS_VERSION(Gps_CNAV_Ephemeris, 2)
00161
00162
00163 /** \} */
00164 /** \} */
00165 #endif  // GNSS_SDR_GPS_CNAV_EPHEMERIS_H
```

13.760 gps_cnav_iono.h File Reference

Interface of a GPS CNAV IONOSPHERIC MODEL storage.
`#include "gps_iono.h"`

Classes

- class Gps_CNAV_Iono

  This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.

13.760.1 Detailed Description

Interface of a GPS CNAV IONOSPHERIC MODEL storage.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_cnav_iono.h.

13.761 gps_cnav_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_cnav_iono.h
00003  * \brief  Interface of a GPS CNAV IONOSPHERIC MODEL storage
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_CNAV_IONO_H
00019 #define GNSS_SDR_GPS_CNAV_IONO_H
00020
00021 #include "gps_iono.h"
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M
00031  *
00032  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
00033  */
00034 class Gps_CNAV_Iono : public Gps_Iono
00035 {
00036 public:
```

```
00037     Gps_CNAV_Iono() = default;  //!< Default constructor
00038 };
00039
00040
00041 /** \} */
00042 /** \} */
00043 #endif  // GNSS_SDR_GPS_CNAV_IONO_H
```

13.762 gps_cnav_navigation_message.h File Reference

Interface of a GPS CNAV Data message decoder.

```
#include "GPS_CNAV.h"
#include "gps_cnav_eop.h"
#include "gps_cnav_ephemeris.h"
#include "gps_cnav_iono.h"
#include "gps_cnav_utc_model.h"
#include <bitset>
#include <cstdint>
#include <map>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Gps_CNAV_Navigation_Message

  This class decodes a GPS CNAV Data message as described in IS-GPS-200M.

Enumerations

- enum class **CnavSystem** { **GPS** , **QZSS** }

13.762.1 Detailed Description

Interface of a GPS CNAV Data message decoder.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

Carles Fernandez Prades, 2015-2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_cnav_navigation_message.h.

13.763 gps_cnav_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_cnav_navigation_message.h
00003  * \brief  Interface of a GPS CNAV Data message decoder
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  * \author Carles Fernandez Prades, 2015-2026. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
```

```
00017
00018
00019 #ifndef GNSS_SDR_GPS_CNAV_NAVIGATION_MESSAGE_H
00020 #define GNSS_SDR_GPS_CNAV_NAVIGATION_MESSAGE_H
00021
00022
00023 #include "GPS_CNAV.h"
00024 #include "gps_cnav_eop.h"
00025 #include "gps_cnav_ephemeris.h"
00026 #include "gps_cnav_iono.h"
00027 #include "gps_cnav_utc_model.h"
00028 #include <bitset>
00029 #include <cstdint>
00030 #include <map>
00031 #include <string>
00032 #include <utility>
00033 #include <vector>
00034
00035 /** \addtogroup Core
00036  * \{ */
00037 /** \addtogroup System_Parameters
00038  * \{ */
00039
00040 enum class CnavSystem
00041 {
00042     GPS,
00043     QZSS
00044 };
00045
00046 /*!
00047  * \brief This class decodes a GPS CNAV Data message as described in IS-GPS-200M
00048  *
00049  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
00050  */
00051 class Gps_CNAV_Navigation_Message
00052 {
00053 public:
00054     /*!
00055      * Default constructor
00056      */
00057     explicit Gps_CNAV_Navigation_Message(CnavSystem system = CnavSystem::GPS);
00058
00059     void decode_page(const std::bitset<GPS_CNAV_DATA_PAGE_BITS>& data_bits);
00060
00061     /*!
00062      * \brief Obtain a GPS SV Ephemeris class filled with current SV data
00063      */
00064     Gps_CNAV_Ephemeris get_ephemeris() const;
00065
00066     /*!
00067      * \brief Check if a newly decoded CNAV Earth orientation record is available
00068      */
00069     bool have_new_eop();
00070
00071     /*!
00072      * \brief Obtain the latest CNAV Earth orientation parameters
00073      */
00074     Gps_CNAV_Eop get_eop() const;
00075
00076     /*!
00077      * \brief Check if we have a new iono record stored in the GPS ephemeris class
00078      */
00079     bool have_new_iono();
00080
00081     /*!
00082      * \brief Obtain a GPS ionospheric correction parameters class filled with current SV data
00083      */
00084     Gps_CNAV_Iono get_iono() const;
00085
00086     /*!
00087      * \brief Obtain a GPS UTC model parameters class filled with current SV data
00088      */
00089     Gps_CNAV_Utc_Model get_utc_model();
00090
00091     /*!
00092      * \briefCheck if we have a new GPS UTC model record stored in the GPS ephemeris class
00093      */
00094     bool have_new_utc_model();
00095
00096     /*!
00097      * \brief Check if we have a new ephemeris stored in the GPS ephemeris class
00098      */
00099     bool have_new_ephemeris();
00100
00101 private:
00102     uint64_t read_navigation_unsigned(const std::bitset<GPS_CNAV_DATA_PAGE_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
```

```
00103     int64_t read_navigation_signed(const std::bitset<GPS_CNAV_DATA_PAGE_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00104     bool read_navigation_bool(const std::bitset<GPS_CNAV_DATA_PAGE_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t>& parameter) const;
00105     void decode_clock_fields(const std::bitset<GPS_CNAV_DATA_PAGE_BITS>& data_bits);
00106
00107     Gps_CNAV_Ephemeris ephemeris_record{};
00108     Gps_CNAV_Eop eop_record{};
00109     Gps_CNAV_Iono iono_record{};
00110     Gps_CNAV_Utc_Model utc_model_record{};
00111
00112     std::map<int32_t, std::string> satelliteBlock;  //!< Map that stores to which block the PRN
      belongs https://www.navcen.uscg.gov/?Do=constellationStatus
00113
00114     CnavSystem d_system;
00115     int32_t d_TOW{};
00116     int32_t d_ephemeris_top{};
00117     int32_t d_clock_top{};
00118
00119     bool b_flag_ephemeris_1{};
00120     bool b_flag_ephemeris_2{};
00121     bool b_flag_clock_valid{};
00122     bool b_flag_eop_valid{};
00123     bool b_flag_iono_valid{};  //!< If set, it indicates that the ionospheric parameters are filled
      and are not yet read by the get_iono
00124     bool b_flag_utc_valid{};   //!< If set, it indicates that the utc parameters are filled and are
      not yet read by the get_utc_model
00125 };
00126
00127
00128 /** \} */
00129 /** \} */
00130 #endif  // GNSS_SDR_GPS_CNAV_NAVIGATION_MESSAGE_H
```

13.764 gps_cnav_utc_model.h File Reference

Interface of a GPS CNAV UTC MODEL storage.

```
#include "gps_utc_model.h"
```

Classes

- class Gps_CNAV_Utc_Model

  This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M.

13.764.1 Detailed Description

Interface of a GPS CNAV UTC MODEL storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_cnav_utc_model.h.

13.765 gps_cnav_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_cnav_utc_model.h
00003  * \brief  Interface of a GPS CNAV UTC MODEL storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
```

```
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_CNAV_UTC_MODEL_H
00019 #define GNSS_SDR_GPS_CNAV_UTC_MODEL_H
00020
00021 #include "gps_utc_model.h"
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief This class is a storage for the GPS UTC MODEL data as described in in IS-GPS-200M
00031  *
00032  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix III
00033  */
00034 class Gps_CNAV_Utc_Model : public Gps_Utc_Model
00035 {
00036 public:
00037     Gps_CNAV_Utc_Model() = default;  //!< Default constructor
00038 };
00039
00040
00041 /** \} */
00042 /** \} */
00043 #endif  // GNSS_SDR_GPS_CNAV_UTC_MODEL_H
```

13.766 gps_ephemeris.h File Reference

Interface of a GPS EPHEMERIS storage.

```
#include "gnss_ephemeris.h"
#include <boost/serialization/nvp.hpp>
#include <cstdint>
#include <map>
#include <string>
```

Classes

- class Gps_Ephemeris

  This class is a storage and orbital model functions for the GPS SV ephemeris data as described in IS-GPS-200M.

13.766.1 Detailed Description

Interface of a GPS EPHEMERIS storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_ephemeris.h.

13.767 gps_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_ephemeris.h
00003  * \brief  Interface of a GPS EPHEMERIS storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
```

```
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_EPHEMERIS_H
00019 #define GNSS_SDR_GPS_EPHEMERIS_H
00020
00021
00022 #include "gnss_ephemeris.h"
00023 #include <boost/serialization/nvp.hpp>
00024 #include <cstdint>
00025 #include <map>
00026 #include <string>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033
00034 /*!
00035  * \brief This class is a storage and orbital model functions for the GPS SV
00036  * ephemeris data as described in IS-GPS-200M
00037  *
00038  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
00039  */
00040 class Gps_Ephemeris : public Gnss_Ephemeris
00041 {
00042 public:
00043     /*!
00044      * Default constructor
00045      */
00046     Gps_Ephemeris();
00047
00048     void set_system(char system)
00049     {
00050         this->System = system;
00051     }
00052
00053     char get_system() const
00054     {
00055         return this->System;
00056     }
00057
00058     int32_t code_on_L2{};   //!< If 1, P code ON in L2;  if 2, C/A code ON in L2;
00059     bool L2_P_data_flag{};  //!< When true, indicates that the NAV data stream was commanded OFF on
      the P-code of the L2 channel
00060     int32_t SV_accuracy{};  //!< User Range Accuracy (URA) index of the SV (reference paragraph 6.2.1)
      for the standard positioning service user (Ref 20.3.3.3.1.3 IS-GPS-200M)
00061     int32_t SV_health{};    //!< Satellite heath status
00062     double TGD{};           //!< Estimated Group Delay Differential: L1-L2 correction term only for
      the benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
00063     int32_t IODC{};         //!< Issue of Data, Clock
00064     int32_t IODE_SF2{};     //!< Issue of Data, Ephemeris (IODE), subframe 2
00065     int32_t IODE_SF3{};     //!< Issue of Data, Ephemeris (IODE), subframe 3
00066     int32_t AODO{};         //!< Age of Data Offset (AODO) term for the navigation message correction
      table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s]
00067
00068     bool fit_interval_flag{};  //!< indicates the curve-fit interval used by the CS (Block
      II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4
      hours, 1 = greater than 4 hours.
00069     double spare1{};
00070     double spare2{};
00071
00072     // Flags
00073
00074     /*! \brief If true, enhanced level of integrity assurance.
00075      *
00076      *  If false, indicates that the conveying signal is provided with the legacy level of integrity
      assurance.
00077      *  That is, the probability that the instantaneous URE of the conveying signal exceeds 4.42 times
      the upper bound
00078      *  value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying
      alert, is less
00079      *  than 1E-5 per hour. If true, indicates that the conveying signal is provided with an enhanced
      level of
00080      *  integrity assurance. That is, the probability that the instantaneous URE of the conveying
      signal exceeds 5.73
00081      *  times the upper bound value of the current broadcast URA index, for more than 5.2 seconds,
      without an
00082      *  accompanying alert, is less than 1E-8 per hour.
00083      */
00084     bool integrity_status_flag{};
00085     bool alert_flag{};         //!< If true, indicates that the SV URA may be worse than indicated in
```

```
    d_SV_accuracy, use that SV at our own risk.
00086     bool antispoofing_flag{};  //!< If true, the AntiSpoofing mode is ON in that SV
00087
00088     std::map<int, std::string> satelliteBlock;  //!< Map that stores to which block the PRN belongs
    https://www.navcen.uscg.gov/?Do=constellationStatus
00089
00090     template <class Archive>
00091
00092     /*!
00093      * \brief Serialize is a boost standard method to be called by the boost XML
00094      * serialization. Here is used to save the ephemeris data on disk file.
00095      */
00096     inline void serialize(Archive& archive, const uint32_t version)
00097     {
00098         using boost::serialization::make_nvp;
00099         if (version)
00100             {
00101             };
00102
00103         archive& BOOST_SERIALIZATION_NVP(PRN);
00104         archive& BOOST_SERIALIZATION_NVP(M_0);
00105         archive& BOOST_SERIALIZATION_NVP(delta_n);
00106         archive& BOOST_SERIALIZATION_NVP(ecc);
00107         archive& BOOST_SERIALIZATION_NVP(sqrtA);
00108         archive& BOOST_SERIALIZATION_NVP(OMEGA_0);
00109         archive& BOOST_SERIALIZATION_NVP(i_0);
00110         archive& BOOST_SERIALIZATION_NVP(omega);
00111         archive& BOOST_SERIALIZATION_NVP(OMEGAdot);
00112         archive& BOOST_SERIALIZATION_NVP(idot);
00113         archive& BOOST_SERIALIZATION_NVP(Cuc);
00114         archive& BOOST_SERIALIZATION_NVP(Cus);
00115         archive& BOOST_SERIALIZATION_NVP(Crc);
00116         archive& BOOST_SERIALIZATION_NVP(Crs);
00117         archive& BOOST_SERIALIZATION_NVP(Cic);
00118         archive& BOOST_SERIALIZATION_NVP(Cis);
00119         archive& BOOST_SERIALIZATION_NVP(toe);
00120         archive& BOOST_SERIALIZATION_NVP(toc);
00121         archive& BOOST_SERIALIZATION_NVP(af0);
00122         archive& BOOST_SERIALIZATION_NVP(af1);
00123         archive& BOOST_SERIALIZATION_NVP(af2);
00124         archive& BOOST_SERIALIZATION_NVP(WN);
00125         archive& BOOST_SERIALIZATION_NVP(tow);
00126         archive& BOOST_SERIALIZATION_NVP(satClkDrift);
00127         archive& BOOST_SERIALIZATION_NVP(dtr);
00128
00129         archive& BOOST_SERIALIZATION_NVP(IODE_SF2);
00130         archive& BOOST_SERIALIZATION_NVP(IODE_SF3);
00131         archive& BOOST_SERIALIZATION_NVP(code_on_L2);
00132         archive& BOOST_SERIALIZATION_NVP(L2_P_data_flag);
00133         archive& BOOST_SERIALIZATION_NVP(SV_accuracy);
00134         archive& BOOST_SERIALIZATION_NVP(SV_health);
00135         archive& BOOST_SERIALIZATION_NVP(TGD);
00136         archive& BOOST_SERIALIZATION_NVP(IODC);
00137         archive& BOOST_SERIALIZATION_NVP(AODO);
00138         archive& BOOST_SERIALIZATION_NVP(fit_interval_flag);
00139         archive& BOOST_SERIALIZATION_NVP(spare1);
00140         archive& BOOST_SERIALIZATION_NVP(spare2);
00141         archive& BOOST_SERIALIZATION_NVP(integrity_status_flag);
00142         archive& BOOST_SERIALIZATION_NVP(alert_flag);
00143         archive& BOOST_SERIALIZATION_NVP(antispoofing_flag);
00144     }
00145 };
00146
00147
00148 /** \} */
00149 /** \} */
00150 #endif  // GNSS_SDR_GPS_EPHEMERIS_H
```

13.768 gps_iono.h File Reference

Interface of a GPS IONOSPHERIC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
```

Classes

- class Gps_Iono

  This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M.

13.768.1 Detailed Description

Interface of a GPS IONOSPHERIC MODEL storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_iono.h.

13.769 gps_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_iono.h
00003  * \brief  Interface of a GPS IONOSPHERIC MODEL storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_IONO_H
00019 #define GNSS_SDR_GPS_IONO_H
00020
00021
00022 #include <boost/serialization/nvp.hpp>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class is a storage for the GPS IONOSPHERIC data as described in IS-GPS-200M
00032  *
00033  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
00034  */
00035 class Gps_Iono
00036 {
00037 public:
00038     Gps_Iono() = default;  //!< Default constructor
00039
00040     // Ionospheric parameters
00041     double alpha0{};  //!< Coefficient 0 of a cubic equation representing the amplitude of the
      vertical delay [s]
00042     double alpha1{};  //!< Coefficient 1 of a cubic equation representing the amplitude of the
      vertical delay [s/semi-circle]
00043     double alpha2{};  //!< Coefficient 2 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^2]
00044     double alpha3{};  //!< Coefficient 3 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^3]
00045     double beta0{};   //!< Coefficient 0 of a cubic equation representing the period of the model [s]
00046     double beta1{};   //!< Coefficient 1 of a cubic equation representing the period of the model
      [s/semi-circle]
00047     double beta2{};   //!< Coefficient 2 of a cubic equation representing the period of the model
      [s(semi-circle)^2]
00048     double beta3{};   //!< Coefficient 3 of a cubic equation representing the period of the model
      [s(semi-circle)^3]
00049
00050     bool valid{};  //!< Valid flag
00051
00052     template <class Archive>
00053
00054     /*!
00055      * \brief Serialize is a boost standard method to be called by the boost XML
00056      * serialization. Here is used to save the ephemeris data on disk file.
00057      */
00058     inline void serialize(Archive& archive, const unsigned int version)
00059     {
00060         if (version)
```

```
00061             {
00062             };
00063         archive& BOOST_SERIALIZATION_NVP(alpha0);
00064         archive& BOOST_SERIALIZATION_NVP(alpha1);
00065         archive& BOOST_SERIALIZATION_NVP(alpha2);
00066         archive& BOOST_SERIALIZATION_NVP(alpha3);
00067         archive& BOOST_SERIALIZATION_NVP(beta0);
00068         archive& BOOST_SERIALIZATION_NVP(beta1);
00069         archive& BOOST_SERIALIZATION_NVP(beta2);
00070         archive& BOOST_SERIALIZATION_NVP(beta3);
00071     }
00072 };
00073
00074
00075 /** \} */
00076 /** \} */
00077 #endif  // GNSS_SDR_GPS_IONO_H
```

13.770 GPS_L1_CA.h File Reference

Defines system parameters for GPS L1 C/A signal and NAV data.

```
#include "MATH_CONSTANTS.h"
#include "gnss_frequencies.h"
#include <cstdint>
#include <utility>
#include <vector>
```

Functions

- const std::vector< std::pair< int32_t, int32_t > > **TOW** ({{31, 17}})
- const std::vector< std::pair< int32_t, int32_t > > **INTEGRITY_STATUS_FLAG** ({{23, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ALERT_FLAG** ({{48, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **ANTI_SPOOFING_FLAG** ({{49, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **SUBFRAME_ID** ({{50, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **GPS_WEEK** ({{61, 10}})
- const std::vector< std::pair< int32_t, int32_t > > **CA_OR_P_ON_L2** ({{71, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_ACCURACY** ({{73, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_HEALTH** ({{77, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **L2_P_DATA_FLAG** ({{91, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **T_GD** ({{197, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **IODC** ({{83, 2}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OC** ({{219, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F2** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F1** ({{249, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **A_F0** ({{271, 22}})
- const std::vector< std::pair< int32_t, int32_t > > **IODE_SF2** ({{61, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **C_RS** ({{69, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTA_N** ({{91, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **M_0** ({{107, 8}, {121, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_UC** ({{151, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ECCENTRICITY** ({{167, 8}, {181, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_US** ({{211, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **SQRT_A** ({{227, 8}, {241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OE** ({{271, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **FIT_INTERVAL_FLAG** ({{287, 1}})
- const std::vector< std::pair< int32_t, int32_t > > **AODO** ({{288, 5}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IC** ({{61, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_0** ({{77, 8}, {91, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **C_IS** ({{121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **I_0** ({{137, 8}, {151, 24}})

- const std::vector< std::pair< int32_t, int32_t > > **C_RC** ({{181, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA** ({{197, 8}, {211, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **OMEGA_DOT** ({{241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **IODE_SF3** ({{271, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **I_DOT** ({{279, 14}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_DATA_ID** ({{61, 2}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_PAGE** ({{63, 6}})
- const std::vector< std::pair< int32_t, int32_t > > ALPHA_0 ({{69, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_1** ({{77, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_2** ({{91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALPHA_3** ({{99, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_0** ({{107, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_1** ({{121, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_2** ({{129, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **BETA_3** ({{137, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **A_1** ({{151, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **A_0** ({{181, 24}, {211, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **T_OT** ({{219, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_T** ({{227, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTAT_LS** ({{241, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_LSF** ({{249, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DN** ({{257, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **DELTAT_LSF** ({{271, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV1** ({{69, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV2** ({{73, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV3** ({{77, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV4** ({{81, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV5** ({{91, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV6** ({{95, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV7** ({{99, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV8** ({{103, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV9** ({{107, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV10** ({{111, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV11** ({{121, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV12** ({{125, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV13** ({{129, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV14** ({{133, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV15** ({{137, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV16** ({{141, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV17** ({{151, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV18** ({{155, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV19** ({{159, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV20** ({{163, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV21** ({{167, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV22** ({{171, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV23** ({{181, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV24** ({{185, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV25** ({{189, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV26** ({{193, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV27** ({{197, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV28** ({{201, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV29** ({{211, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV30** ({{215, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV31** ({{219, 4}})
- const std::vector< std::pair< int32_t, int32_t > > **SV_CONFIG_SV32** ({{223, 4}})

- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV25** ({{229, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV26** ({{241, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV27** ({{247, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV28** ({{253, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV29** ({{259, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV30** ({{271, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV31** ({{277, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV32** ({{283, 6}})
- const std::vector< std::pair< int32_t, int32_t > > T_OA ({{69, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **WN_A** ({{77, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV1** ({{91, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV2** ({{97, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV3** ({{103, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV4** ({{109, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV5** ({{121, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV6** ({{127, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV7** ({{133, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV8** ({{139, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV9** ({{151, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV10** ({{157, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV11** ({{163, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV12** ({{169, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV13** ({{181, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV14** ({{187, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV15** ({{193, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV16** ({{199, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV17** ({{211, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV18** ({{217, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV19** ({{223, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV20** ({{229, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV21** ({{241, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV22** ({{247, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV23** ({{253, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **HEALTH_SV24** ({{259, 6}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_ECC** ({{69, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_TOA** ({{91, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_DELTAI** ({{99, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGADOT** ({{121, 16}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_SVHEALTH** ({{137, 8}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_SQUAREA** ({{151, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGAZERO** ({{181, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_OMEGA** ({{211, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_MZERO** ({{241, 24}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_AF0** ({{271, 8}, {290, 3}})
- const std::vector< std::pair< int32_t, int32_t > > **ALM_AF1** ({{279, 11}})

Variables

- constexpr double GPS_L1_FREQ_HZ = FREQ1

  L1 [Hz].
- constexpr double GPS_L1_CA_CODE_RATE_CPS = 1.023e6

  GPS L1 C/A code rate [chips/s].
- constexpr double GPS_L1_CA_CODE_LENGTH_CHIPS = 1023.0

  GPS L1 C/A code length [chips].

- constexpr double GPS_L1_CA_CODE_PERIOD_S = 0.001

  GPS L1 C/A code period [seconds].
- constexpr double GPS_L1_CA_CHIP_PERIOD_S = 9.7752e-07

  GPS L1 C/A chip period [seconds].
- constexpr uint32_t GPS_L1_CA_CODE_PERIOD_MS = 1U

  GPS L1 C/A code period [ms].
- constexpr uint32_t GPS_L1_CA_BIT_PERIOD_MS = 20U

  GPS L1 C/A bit period [ms].
- constexpr double MAX_TOA_DELAY_MS = 20.0

  Maximum Time-Of-Arrival (TOA) difference between satellites for a receiver operated on Earth surface is 20 ms.
- constexpr uint32_t GPS_L1_CA_OPT_ACQ_FS_SPS = 2000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L1_CA_HISTORY_DEEP = 100
- constexpr double GPS_CA_PREAMBLE_DURATION_S = 0.160
- constexpr int32_t GPS_CA_PREAMBLE_LENGTH_BITS = 8
- constexpr int32_t GPS_CA_PREAMBLE_LENGTH_SYMBOLS = 160
- constexpr int32_t GPS_CA_PREAMBLE_DURATION_MS = 160
- constexpr int32_t GPS_CA_TELEMETRY_RATE_BITS_SECOND = 50

  NAV message bit rate [bits/s].
- constexpr int32_t GPS_CA_TELEMETRY_SYMBOLS_PER_BIT = 20
- constexpr int32_t GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND = GPS_CA_TELEMETRY_RATE_BITS_SECOND ∗ GPS_CA_TELEMETRY_SYMBOLS_PER_BIT

  NAV message bit rate [symbols/s].
- constexpr int32_t GPS_WORD_LENGTH = 4

  CRC + GPS WORD (-2 -1 0 ... 29) Bits = 4 bytes.
- constexpr int32_t GPS_SUBFRAME_LENGTH = 40

  GPS_WORD_LENGTH x 10 = 40 bytes.
- constexpr int32_t GPS_SUBFRAME_BITS = 300

  Number of bits per subframe in the NAV message [bits].
- constexpr int32_t GPS_SUBFRAME_SECONDS = 6

  Subframe duration [seconds].
- constexpr int32_t GPS_SUBFRAME_MS = 6000

  Subframe duration [seconds].
- constexpr int32_t GPS_WORD_BITS = 30

  Number of bits per word in the NAV message [bits].
- constexpr int32_t GPS_LNAV_DATA_ID = 1

  IS-GPS-200N LNAV Data ID number 2, encoded as binary 01.
- constexpr char GPS_CA_PREAMBLE [9] = "10001011"
- constexpr char GPS_CA_PREAMBLE_SYMBOLS_STR [161] = "11111111111111111111000000000000000000000000000000
- constexpr double T_GD_LSB = TWO_N31
- constexpr int32_t T_OC_LSB = static_cast<int32_t>(TWO_P4)
- constexpr double A_F2_LSB = TWO_N55
- constexpr double A_F1_LSB = TWO_N43
- constexpr double A_F0_LSB = TWO_N31
- constexpr double C_RS_LSB = TWO_N5
- constexpr double DELTA_N_LSB = PI_TWO_N43
- constexpr double M_0_LSB = PI_TWO_N31
- constexpr double C_UC_LSB = TWO_N29
- constexpr double ECCENTRICITY_LSB = TWO_N33
- constexpr double C_US_LSB = TWO_N29
- constexpr double SQRT_A_LSB = TWO_N19
- constexpr int32_t T_OE_LSB = static_cast<int32_t>(TWO_P4)

- constexpr int32_t AODO_LSB = 900
- constexpr double C_IC_LSB = TWO_N29
- constexpr double OMEGA_0_LSB = PI_TWO_N31
- constexpr double C_IS_LSB = TWO_N29
- constexpr double I_0_LSB = PI_TWO_N31
- constexpr double C_RC_LSB = TWO_N5
- constexpr double OMEGA_LSB = PI_TWO_N31
- constexpr double OMEGA_DOT_LSB = PI_TWO_N43
- constexpr double I_DOT_LSB = PI_TWO_N43
- constexpr double ALPHA_0_LSB = TWO_N30
- constexpr double ALPHA_1_LSB = TWO_N27
- constexpr double ALPHA_2_LSB = TWO_N24
- constexpr double ALPHA_3_LSB = TWO_N24
- constexpr double BETA_0_LSB = TWO_P11
- constexpr double BETA_1_LSB = TWO_P14
- constexpr double BETA_2_LSB = TWO_P16
- constexpr double BETA_3_LSB = TWO_P16
- constexpr double A_1_LSB = TWO_N50
- constexpr double A_0_LSB = TWO_N30
- constexpr double T_OT_LSB = TWO_P12
- constexpr double WN_T_LSB = 1
- constexpr double DELTAT_LS_LSB = 1
- constexpr double WN_LSF_LSB = 1
- constexpr double DN_LSB = 1
- constexpr double DELTAT_LSF_LSB = 1
- constexpr int32_t GPS_SV_CONFIG_AS_FLAG_MASK = 0x8

  A-S flag (1 = ON).
- constexpr int32_t GPS_SV_CONFIG_CODE_MASK = 0x7

  SV configuration.
- constexpr int32_t GPS_SV_CONFIG_UNKNOWN = 0

  000: no information
- constexpr int32_t GPS_SV_CONFIG_BLOCK_IIR_M = 2

  010: first with L2C
- constexpr int32_t GPS_SV_CONFIG_BLOCK_IIF = 3

  011: first with L5
- constexpr int32_t T_OA_LSB = TWO_P12
- constexpr double ALM_ECC_LSB = TWO_N21
- constexpr int32_t ALM_TOA_LSB = static_cast<int32_t>(TWO_P12)
- constexpr double ALM_DELTAI_LSB = TWO_N19
- constexpr double ALM_OMEGADOT_LSB = TWO_N38
- constexpr double ALM_SQUAREA_LSB = TWO_N11
- constexpr double ALM_OMEGAZERO_LSB = TWO_N23
- constexpr double ALM_OMEGA_LSB = TWO_N23
- constexpr double ALM_MZERO_LSB = TWO_N23
- constexpr double ALM_AF0_LSB = TWO_N20
- constexpr double ALM_AF1_LSB = TWO_N38

13.770.1 Detailed Description

Defines system parameters for GPS L1 C/A signal and NAV data.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file GPS_L1_CA.h.

13.771 GPS_L1_CA.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file GPS_L1_CA.h
00003  * \brief  Defines system parameters for GPS L1 C/A signal and NAV data
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_L1_CA_H
00019 #define GNSS_SDR_GPS_L1_CA_H
00020
00021 #include "MATH_CONSTANTS.h"
00022 #include "gnss_frequencies.h"
00023 #include <cstdint>
00024 #include <utility>  // std::pair
00025 #include <vector>
00026
00027 /** \addtogroup Core
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032
00033 // carrier and code frequencies
00034 constexpr double GPS_L1_FREQ_HZ = FREQ1;                //!< L1 [Hz]
00035 constexpr double GPS_L1_CA_CODE_RATE_CPS = 1.023e6;     //!< GPS L1 C/A code rate [chips/s]
00036 constexpr double GPS_L1_CA_CODE_LENGTH_CHIPS = 1023.0;  //!< GPS L1 C/A code length [chips]
00037 constexpr double GPS_L1_CA_CODE_PERIOD_S = 0.001;       //!< GPS L1 C/A code period [seconds]
00038 constexpr double GPS_L1_CA_CHIP_PERIOD_S = 9.7752e-07;  //!< GPS L1 C/A chip period [seconds]
00039 constexpr uint32_t GPS_L1_CA_CODE_PERIOD_MS = 1U;       //!< GPS L1 C/A code period [ms]
00040 constexpr uint32_t GPS_L1_CA_BIT_PERIOD_MS = 20U;       //!< GPS L1 C/A bit period [ms]
00041
00042 /*!
00043  * \brief Maximum Time-Of-Arrival (TOA) difference between satellites for a receiver operated on Earth surface is 20 ms
00044  *
00045  * According to the GPS orbit model described in [1] Pag. 32.
00046  * It should be taken into account to set the buffer size for the PRN start timestamp in the pseudoranges block.
00047  * [1] J. Bao-Yen Tsui, Fundamentals of Global Positioning System Receivers. A Software Approach, John Wiley & Sons,
00048  * Inc., Hoboken, NJ, 2nd edition, 2005.
00049  */
00050 constexpr double MAX_TOA_DELAY_MS = 20.0;
00051
00052 // optimum parameters
00053 constexpr uint32_t GPS_L1_CA_OPT_ACQ_FS_SPS = 2000000;  //!< Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate
00054
00055 // OBSERVABLE HISTORY DEEP FOR INTERPOLATION
00056 constexpr int32_t GPS_L1_CA_HISTORY_DEEP = 100;
00057
00058 // NAVIGATION MESSAGE DEMODULATION AND DECODING
00059 constexpr double GPS_CA_PREAMBLE_DURATION_S = 0.160;
00060 constexpr int32_t GPS_CA_PREAMBLE_LENGTH_BITS = 8;
00061 constexpr int32_t GPS_CA_PREAMBLE_LENGTH_SYMBOLS = 160;
00062 constexpr int32_t GPS_CA_PREAMBLE_DURATION_MS = 160;
00063 constexpr int32_t GPS_CA_TELEMETRY_RATE_BITS_SECOND = 50;  //!< NAV message bit rate [bits/s]
00064 constexpr int32_t GPS_CA_TELEMETRY_SYMBOLS_PER_BIT = 20;
00065 constexpr int32_t GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND = GPS_CA_TELEMETRY_RATE_BITS_SECOND * GPS_CA_TELEMETRY_SYMBOLS_PER_BIT;  //!< NAV message bit rate [symbols/s]
00066 constexpr int32_t GPS_WORD_LENGTH = 4;
      //!< CRC + GPS WORD (-2 -1 0 ... 29) Bits = 4 bytes
00067 constexpr int32_t GPS_SUBFRAME_LENGTH = 40;
      //!< GPS_WORD_LENGTH x 10 = 40 bytes
00068 constexpr int32_t GPS_SUBFRAME_BITS = 300;
      //!< Number of bits per subframe in the NAV message [bits]
00069 constexpr int32_t GPS_SUBFRAME_SECONDS = 6;
      //!< Subframe duration [seconds]
00070 constexpr int32_t GPS_SUBFRAME_MS = 6000;
      //!< Subframe duration [seconds]
00071 constexpr int32_t GPS_WORD_BITS = 30;
      //!< Number of bits per word in the NAV message [bits]
00072 constexpr int32_t GPS_LNAV_DATA_ID = 1;
```

```
      //!< IS-GPS-200N LNAV Data ID number 2, encoded as binary 01.
00073 constexpr char GPS_CA_PREAMBLE[9] = "10001011";
00074 constexpr char GPS_CA_PREAMBLE_SYMBOLS_STR[161] =
      "11111111111111111111000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111000000000000000000
00075
00076 // GPS NAVIGATION MESSAGE STRUCTURE
00077 // NAVIGATION MESSAGE FIELDS POSITIONS (from IS-GPS-200M Appendix II)
00078
00079 // SUBFRAME 1-5 (TLM and HOW)
00080
00081 const std::vector<std::pair<int32_t, int32_t» TOW({{31, 17}});
00082 const std::vector<std::pair<int32_t, int32_t» INTEGRITY_STATUS_FLAG({{23, 1}});
00083 const std::vector<std::pair<int32_t, int32_t» ALERT_FLAG({{48, 1}});
00084 const std::vector<std::pair<int32_t, int32_t» ANTI_SPOOFING_FLAG({{49, 1}});
00085 const std::vector<std::pair<int32_t, int32_t» SUBFRAME_ID({{50, 3}});
00086
00087 // SUBFRAME 1
00088 const std::vector<std::pair<int32_t, int32_t» GPS_WEEK({{61, 10}});
00089 const std::vector<std::pair<int32_t, int32_t» CA_OR_P_ON_L2({{71, 2}});  //*
00090 const std::vector<std::pair<int32_t, int32_t» SV_ACCURACY({{73, 4}});
00091 const std::vector<std::pair<int32_t, int32_t» SV_HEALTH({{77, 6}});
00092 const std::vector<std::pair<int32_t, int32_t» L2_P_DATA_FLAG({{91, 1}});
00093 const std::vector<std::pair<int32_t, int32_t» T_GD({{197, 8}});
00094 constexpr double T_GD_LSB = TWO_N31;
00095 const std::vector<std::pair<int32_t, int32_t» IODC({{83, 2}, {211, 8}});
00096 const std::vector<std::pair<int32_t, int32_t» T_OC({{219, 16}});
00097 constexpr int32_t T_OC_LSB = static_cast<int32_t>(TWO_P4);
00098 const std::vector<std::pair<int32_t, int32_t» A_F2({{241, 8}});
00099 constexpr double A_F2_LSB = TWO_N55;
00100 const std::vector<std::pair<int32_t, int32_t» A_F1({{249, 16}});
00101 constexpr double A_F1_LSB = TWO_N43;
00102 const std::vector<std::pair<int32_t, int32_t» A_F0({{271, 22}});
00103 constexpr double A_F0_LSB = TWO_N31;
00104
00105 // SUBFRAME 2
00106 const std::vector<std::pair<int32_t, int32_t» IODE_SF2({{61, 8}});
00107 const std::vector<std::pair<int32_t, int32_t» C_RS({{69, 16}});
00108 constexpr double C_RS_LSB = TWO_N5;
00109 const std::vector<std::pair<int32_t, int32_t» DELTA_N({{91, 16}});
00110 constexpr double DELTA_N_LSB = PI_TWO_N43;
00111 const std::vector<std::pair<int32_t, int32_t» M_0({{107, 8}, {121, 24}});
00112 constexpr double M_0_LSB = PI_TWO_N31;
00113 const std::vector<std::pair<int32_t, int32_t» C_UC({{151, 16}});
00114 constexpr double C_UC_LSB = TWO_N29;
00115 const std::vector<std::pair<int32_t, int32_t» ECCENTRICITY({{167, 8}, {181, 24}});
00116 constexpr double ECCENTRICITY_LSB = TWO_N33;
00117 const std::vector<std::pair<int32_t, int32_t» C_US({{211, 16}});
00118 constexpr double C_US_LSB = TWO_N29;
00119 const std::vector<std::pair<int32_t, int32_t» SQRT_A({{227, 8}, {241, 24}});
00120 constexpr double SQRT_A_LSB = TWO_N19;
00121 const std::vector<std::pair<int32_t, int32_t» T_OE({{271, 16}});
00122 constexpr int32_t T_OE_LSB = static_cast<int32_t>(TWO_P4);
00123 const std::vector<std::pair<int32_t, int32_t» FIT_INTERVAL_FLAG({{287, 1}});
00124 const std::vector<std::pair<int32_t, int32_t» AODO({{288, 5}});
00125 constexpr int32_t AODO_LSB = 900;
00126
00127 // SUBFRAME 3
00128 const std::vector<std::pair<int32_t, int32_t» C_IC({{61, 16}});
00129 constexpr double C_IC_LSB = TWO_N29;
00130 const std::vector<std::pair<int32_t, int32_t» OMEGA_0({{77, 8}, {91, 24}});
00131 constexpr double OMEGA_0_LSB = PI_TWO_N31;
00132 const std::vector<std::pair<int32_t, int32_t» C_IS({{121, 16}});
00133 constexpr double C_IS_LSB = TWO_N29;
00134 const std::vector<std::pair<int32_t, int32_t» I_0({{137, 8}, {151, 24}});
00135 constexpr double I_0_LSB = PI_TWO_N31;
00136 const std::vector<std::pair<int32_t, int32_t» C_RC({{181, 16}});
00137 constexpr double C_RC_LSB = TWO_N5;
00138 const std::vector<std::pair<int32_t, int32_t» OMEGA({{197, 8}, {211, 24}});
00139 constexpr double OMEGA_LSB = PI_TWO_N31;
00140 const std::vector<std::pair<int32_t, int32_t» OMEGA_DOT({{241, 24}});
00141 constexpr double OMEGA_DOT_LSB = PI_TWO_N43;
00142 const std::vector<std::pair<int32_t, int32_t» IODE_SF3({{271, 8}});
00143 const std::vector<std::pair<int32_t, int32_t» I_DOT({{279, 14}});
00144 constexpr double I_DOT_LSB = PI_TWO_N43;
00145
00146 // SUBFRAME 4-5
00147 const std::vector<std::pair<int32_t, int32_t» SV_DATA_ID({{61, 2}});
00148 const std::vector<std::pair<int32_t, int32_t» SV_PAGE({{63, 6}});
00149
00150 // SUBFRAME 4
00151 //! \todo read all pages of subframe 4
00152 // Page 18 - Ionospheric and UTC data
00153 const std::vector<std::pair<int32_t, int32_t» ALPHA_0({{69, 8}});
00154 constexpr double ALPHA_0_LSB = TWO_N30;
00155 const std::vector<std::pair<int32_t, int32_t» ALPHA_1({{77, 8}});
00156 constexpr double ALPHA_1_LSB = TWO_N27;
00157 const std::vector<std::pair<int32_t, int32_t» ALPHA_2({{91, 8}});
```

```
00158 constexpr double ALPHA_2_LSB = TWO_N24;
00159 const std::vector<std::pair<int32_t, int32_t» ALPHA_3({{99, 8}});
00160 constexpr double ALPHA_3_LSB = TWO_N24;
00161 const std::vector<std::pair<int32_t, int32_t» BETA_0({{107, 8}});
00162 constexpr double BETA_0_LSB = TWO_P11;
00163 const std::vector<std::pair<int32_t, int32_t» BETA_1({{121, 8}});
00164 constexpr double BETA_1_LSB = TWO_P14;
00165 const std::vector<std::pair<int32_t, int32_t» BETA_2({{129, 8}});
00166 constexpr double BETA_2_LSB = TWO_P16;
00167 const std::vector<std::pair<int32_t, int32_t» BETA_3({{137, 8}});
00168 constexpr double BETA_3_LSB = TWO_P16;
00169 const std::vector<std::pair<int32_t, int32_t» A_1({{151, 24}});
00170 constexpr double A_1_LSB = TWO_N50;
00171 const std::vector<std::pair<int32_t, int32_t» A_0({{181, 24}, {211, 8}});
00172 constexpr double A_0_LSB = TWO_N30;
00173 const std::vector<std::pair<int32_t, int32_t» T_OT({{219, 8}});
00174 constexpr double T_OT_LSB = TWO_P12;
00175 const std::vector<std::pair<int32_t, int32_t» WN_T({{227, 8}});
00176 constexpr double WN_T_LSB = 1;
00177 const std::vector<std::pair<int32_t, int32_t» DELTAT_LS({{241, 8}});
00178 constexpr double DELTAT_LS_LSB = 1;
00179 const std::vector<std::pair<int32_t, int32_t» WN_LSF({{249, 8}});
00180 constexpr double WN_LSF_LSB = 1;
00181 const std::vector<std::pair<int32_t, int32_t» DN({{257, 8}});
00182 constexpr double DN_LSB = 1;
00183 const std::vector<std::pair<int32_t, int32_t» DELTAT_LSF({{271, 8}});
00184 constexpr double DELTAT_LSF_LSB = 1;
00185
00186 // Page 25 - Antispoofing, SV config and SV health (PRN 25 -32)
00187 // A-S flag and SV configuration, 4 bits per SV (IS-GPS-200, 20.3.3.5.1.4).
00188 // MSB: A-S flag (1 = ON). Three LSBs: SV configuration (000 = no info,
00189 // 001 = II/IIA/IIR, 010 = IIR-M, 011 = IIF, 100 = GPS III, others reserved).
00190 constexpr int32_t GPS_SV_CONFIG_AS_FLAG_MASK = 0x8;  //!< A-S flag (1 = ON)
00191 constexpr int32_t GPS_SV_CONFIG_CODE_MASK = 0x7;     //!< SV configuration
00192 constexpr int32_t GPS_SV_CONFIG_UNKNOWN = 0;         //!< 000: no information
00193 constexpr int32_t GPS_SV_CONFIG_BLOCK_IIR_M = 2;     //!< 010: first with L2C
00194 constexpr int32_t GPS_SV_CONFIG_BLOCK_IIF = 3;       //!< 011: first with L5
00195
00196 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV1({{69, 4}});
00197 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV2({{73, 4}});
00198 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV3({{77, 4}});
00199 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV4({{81, 4}});
00200 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV5({{91, 4}});
00201 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV6({{95, 4}});
00202 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV7({{99, 4}});
00203 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV8({{103, 4}});
00204 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV9({{107, 4}});
00205 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV10({{111, 4}});
00206 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV11({{121, 4}});
00207 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV12({{125, 4}});
00208 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV13({{129, 4}});
00209 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV14({{133, 4}});
00210 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV15({{137, 4}});
00211 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV16({{141, 4}});
00212 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV17({{151, 4}});
00213 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV18({{155, 4}});
00214 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV19({{159, 4}});
00215 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV20({{163, 4}});
00216 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV21({{167, 4}});
00217 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV22({{171, 4}});
00218 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV23({{181, 4}});
00219 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV24({{185, 4}});
00220 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV25({{189, 4}});
00221 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV26({{193, 4}});
00222 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV27({{197, 4}});
00223 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV28({{201, 4}});
00224 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV29({{211, 4}});
00225 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV30({{215, 4}});
00226 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV31({{219, 4}});
00227 const std::vector<std::pair<int32_t, int32_t» SV_CONFIG_SV32({{223, 4}});
00228
00229 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV25({{229, 6}});
00230 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV26({{241, 6}});
00231 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV27({{247, 6}});
00232 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV28({{253, 6}});
00233 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV29({{259, 6}});
00234 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV30({{271, 6}});
00235 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV31({{277, 6}});
00236 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV32({{283, 6}});
00237
00238
00239 // SUBFRAME 5
00240 //! \todo read all pages of subframe 5
00241
00242 // page 25 - Health (PRN 1 - 24)
00243 const std::vector<std::pair<int32_t, int32_t» T_OA({{69, 8}});
00244 constexpr int32_t T_OA_LSB = TWO_P12;
```

```
00245 const std::vector<std::pair<int32_t, int32_t» WN_A({{77, 8}});
00246 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV1({{91, 6}});
00247 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV2({{97, 6}});
00248 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV3({{103, 6}});
00249 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV4({{109, 6}});
00250 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV5({{121, 6}});
00251 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV6({{127, 6}});
00252 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV7({{133, 6}});
00253 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV8({{139, 6}});
00254 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV9({{151, 6}});
00255 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV10({{157, 6}});
00256 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV11({{163, 6}});
00257 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV12({{169, 6}});
00258 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV13({{181, 6}});
00259 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV14({{187, 6}});
00260 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV15({{193, 6}});
00261 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV16({{199, 6}});
00262 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV17({{211, 6}});
00263 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV18({{217, 6}});
00264 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV19({{223, 6}});
00265 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV20({{229, 6}});
00266 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV21({{241, 6}});
00267 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV22({{247, 6}});
00268 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV23({{253, 6}});
00269 const std::vector<std::pair<int32_t, int32_t» HEALTH_SV24({{259, 6}});
00270
00271
00272 // Almanac
00273 const std::vector<std::pair<int32_t, int32_t» ALM_ECC({{69, 16}});
00274 constexpr double ALM_ECC_LSB = TWO_N21;
00275 const std::vector<std::pair<int32_t, int32_t» ALM_TOA({{91, 8}});
00276 constexpr int32_t ALM_TOA_LSB = static_cast<int32_t>(TWO_P12);
00277 const std::vector<std::pair<int32_t, int32_t» ALM_DELTAI({{99, 16}});
00278 constexpr double ALM_DELTAI_LSB = TWO_N19;
00279 const std::vector<std::pair<int32_t, int32_t» ALM_OMEGADOT({{121, 16}});
00280 constexpr double ALM_OMEGADOT_LSB = TWO_N38;
00281 const std::vector<std::pair<int32_t, int32_t» ALM_SVHEALTH({{137, 8}});
00282 const std::vector<std::pair<int32_t, int32_t» ALM_SQUAREA({{151, 24}});
00283 constexpr double ALM_SQUAREA_LSB = TWO_N11;
00284 const std::vector<std::pair<int32_t, int32_t» ALM_OMEGAZERO({{181, 24}});
00285 constexpr double ALM_OMEGAZERO_LSB = TWO_N23;
00286 const std::vector<std::pair<int32_t, int32_t» ALM_OMEGA({{211, 24}});
00287 constexpr double ALM_OMEGA_LSB = TWO_N23;
00288 const std::vector<std::pair<int32_t, int32_t» ALM_MZERO({{241, 24}});
00289 constexpr double ALM_MZERO_LSB = TWO_N23;
00290
00291 const std::vector<std::pair<int32_t, int32_t» ALM_AF0({{271, 8}, {290, 3}});
00292 constexpr double ALM_AF0_LSB = TWO_N20;
00293 const std::vector<std::pair<int32_t, int32_t» ALM_AF1({{279, 11}});
00294 constexpr double ALM_AF1_LSB = TWO_N38;
00295 /** \} */
00296 /** \} */
00297 #endif  // GNSS_SDR_GPS_L1_CA_H
```

13.772 GPS_L2C.h File Reference

Defines system parameters for GPS L2C signal.

```
#include "GPS_CNAV.h"
#include "gnss_frequencies.h"
#include <cstdint>
#include <utility>
#include <vector>
```

Variables

- constexpr double GPS_L2_FREQ_HZ = FREQ2

  L2 [Hz].
- constexpr double GPS_L2_L_PERIOD_S = 1.5

  GPS L2 L code period [seconds].
- constexpr double GPS_L2_M_CODE_RATE_CPS = 0.5115e6

  GPS L2 M code rate [chips/s].
- constexpr double GPS_L2_M_CODE_RATE_L_ZEROED_CPS = 1.023e6

  GPS L2 M code rate with L2 M Zeroed [chips/s].

- constexpr double GPS_L2_M_PERIOD_S = 0.02

  GPS L2 M code period [seconds].
- constexpr double GPS_L2_L_CODE_RATE_CPS = 0.5115e6

  GPS L2 L code rate [chips/s].
- constexpr uint32_t GPS_L2_M_CODE_PERIOD_MS = 20

  GPS L2 L code period [ms].
- constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS = 10230

  GPS L2 M code length [chips].
- constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED = 2 * GPS_L2_M_CODE_LENGTH_CHIPS

  GPS L2 M code length with L2 CL zeroed [chips].
- constexpr int32_t GPS_L2_L_CODE_LENGTH_CHIPS = 767250

  GPS L2 L code length [chips].
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_BITS = 300

  GPS L2 CNAV page length, including preamble and CRC [bits].
- constexpr int32_t GPS_L2_SYMBOLS_PER_BIT = 2
- constexpr int32_t GPS_L2_SAMPLES_PER_SYMBOL = 1
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_SYMBOLS = 600
- constexpr int32_t GPS_L2_CNAV_DATA_PAGE_DURATION_S = 12
- constexpr int32_t GPS_L2C_HISTORY_DEEP = 5
- constexpr uint32_t GPS_L2C_OPT_ACQ_FS_SPS = 2000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L2C_M_INIT_REG [115]

13.772.1 Detailed Description

Defines system parameters for GPS L2C signal.

Author

Javier Arribas, 2015. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file GPS_L2C.h.

13.773 GPS_L2C.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file GPS_L2C.h
00003  * \brief  Defines system parameters for GPS L2C signal
00004  * \author Javier Arribas, 2015. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_L2C_H
00019 #define GNSS_SDR_GPS_L2C_H
00020
00021
00022 #include "GPS_CNAV.h"
00023 #include "gnss_frequencies.h"
00024 #include <cstdint>
00025 #include <utility>  // std::pair
00026 #include <vector>
00027
```

```
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033
00034 // carrier and code frequencies
00035 constexpr double GPS_L2_FREQ_HZ = FREQ2;                                          //!< L2 [Hz]
00036 constexpr double GPS_L2_L_PERIOD_S = 1.5;                                         //!< GPS L2 L
      code period [seconds]
00037 constexpr double GPS_L2_M_CODE_RATE_CPS = 0.5115e6;                               //!< GPS L2 M
      code rate [chips/s]
00038 constexpr double GPS_L2_M_CODE_RATE_L_ZEROED_CPS = 1.023e6;                       //!< GPS L2 M
      code rate with L2 M Zeroed [chips/s]
00039 constexpr double GPS_L2_M_PERIOD_S = 0.02;                                       //!< GPS L2 M
      code period [seconds]
00040 constexpr double GPS_L2_L_CODE_RATE_CPS = 0.5115e6;                             //!< GPS L2 L
      code rate [chips/s]
00041 constexpr uint32_t GPS_L2_M_CODE_PERIOD_MS = 20;                                //!<  GPS L2
      L code period [ms]
00042 constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS = 10230;                           //!< GPS L2 M
      code length [chips]
00043 constexpr int32_t GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED = 2 * GPS_L2_M_CODE_LENGTH_CHIPS;  //!< GPS L2 M
      code length with L2 CL zeroed [chips]
00044 constexpr int32_t GPS_L2_L_CODE_LENGTH_CHIPS = 767250;                           //!< GPS L2 L
      code length [chips]
00045 constexpr int32_t GPS_L2_CNAV_DATA_PAGE_BITS = 300;                             //!< GPS L2
      CNAV page length, including preamble and CRC [bits]
00046 constexpr int32_t GPS_L2_SYMBOLS_PER_BIT = 2;
00047 constexpr int32_t GPS_L2_SAMPLES_PER_SYMBOL = 1;
00048 constexpr int32_t GPS_L2_CNAV_DATA_PAGE_SYMBOLS = 600;
00049 constexpr int32_t GPS_L2_CNAV_DATA_PAGE_DURATION_S = 12;
00050
00051 constexpr int32_t GPS_L2C_HISTORY_DEEP = 5;
00052
00053 // optimum parameters
00054 constexpr uint32_t GPS_L2C_OPT_ACQ_FS_SPS = 2000000;  //!< Sampling frequency that maximizes the
      acquisition SNR while using a non-multiple of chip rate
00055
00056 constexpr int32_t GPS_L2C_M_INIT_REG[115] =
00057     {0742417664, 0756014035, 0002747144, 0066265724,     // 1:4
00058         0601403471, 0703232733, 0124510070, 0617316361,  // 5:8
00059         0047541621, 0733031046, 0713512145, 0024437606,
00060         0021264003, 0230655351, 0001314400, 0222021506,
00061         0540264026, 0205521705, 0064022144, 0120161274,
00062         0044023533, 0724744327, 0045743577, 0741201660,
00063         0700274134, 0010247261, 0713433445, 0737324162,
00064         0311627434, 0710452007, 0722462133, 0050172213,
00065         0500653703, 0755077436, 0136717361, 0756675453,
00066         0435506112, 0771353753, 0226107701, 0022025110,
00067         0402466344, 0752566114, 0702011164, 0041216771,
00068         0047457275, 0266333164, 0713167356, 0060546335,
00069         0355173035, 0617201036, 0157465571, 0767360553,
00070         0023127030, 0431343777, 0747317317, 0045706125,
00071         0002744276, 0060036467, 0217744147, 0603340174,  // 57:60
00072         0326616775, 0063240065, 0111460621,              // 61:63
00073         0604055104, 0157065232, 0013305707, 0603552017,  // 159:162
00074         0230461355, 0603653437, 0652346475, 0743107103,
00075         0401521277, 0167335110, 0014013575, 0362051132,
00076         0617753265, 0216363634, 0755561123, 0365304033,
00077         0625025543, 0054420334, 0415473671, 0662364360,
00078         0373446602, 0417564100, 0000526452, 0226631300,
00079         0113752074, 0706134401, 0041352546, 0664630154,
00080         0276524255, 0714720530, 0714051771, 0044526647,
00081         0207164322, 0262120161, 0204244652, 0202133131,
00082         0714351204, 0657127260, 0130567507, 0670517677,
00083         0607275514, 0045413633, 0212645405, 0613700455,
00084         0706202440, 0705056276, 0020373522, 0746013617,
00085         0132720621, 0434015513, 0566721727, 0140633660};
00086
00087
00088 /** \} */
00089 /** \} */
00090 #endif  // GNSS_SDR_GPS_L2C_H
```

13.774 GPS_L5.h File Reference

Defines system parameters for GPS L5 signal.

```
#include "GPS_CNAV.h"
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double GPS_L5_FREQ_HZ = FREQ5

  L5 [Hz].
- constexpr double GPS_L5I_CODE_RATE_CPS = 10.23e6

  GPS L5I code rate [chips/s].
- constexpr double GPS_L5I_PERIOD_S = 0.001

  GPS L5I code period [seconds].
- constexpr double GPS_L5I_SYMBOL_PERIOD_S = 0.01

  GPS L5I symbol period [seconds].
- constexpr double GPS_L5Q_CODE_RATE_CPS = 10.23e6

  GPS L5Q code rate [chips/s].
- constexpr double GPS_L5Q_PERIOD_S = 0.001

  GPS L5Q code period [seconds].
- constexpr int32_t GPS_L5Q_CODE_LENGTH_CHIPS = 10230

  GPS L5Q code length [chips].
- constexpr int32_t GPS_L5I_CODE_LENGTH_CHIPS = 10230

  GPS L5I code length [chips].
- constexpr int32_t GPS_L5I_PERIOD_MS = 1

  GPS L5I code period [ms].
- constexpr int32_t GPS_L5I_SYMBOL_PERIOD_MS = 10

  GPS L5I symbol period [ms].
- constexpr int32_t GPS_L5_HISTORY_DEEP = 5
- constexpr uint32_t GPS_L5_OPT_ACQ_FS_SPS = 10000000

  Sampling frequency that maximizes the acquisition SNR while using a non-multiple of chip rate.
- constexpr int32_t GPS_L5I_INIT_REG [210]
- constexpr int32_t GPS_L5Q_INIT_REG [210]
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_BITS = 300

  GPS L5 CNAV page length, including preamble and CRC [bits].
- constexpr int32_t GPS_L5_SYMBOLS_PER_BIT = 2
- constexpr int32_t GPS_L5_SAMPLES_PER_SYMBOL = 10
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_SYMBOLS = 600
- constexpr int32_t GPS_L5_CNAV_DATA_PAGE_DURATION_S = 6
- constexpr int32_t GPS_L5I_NH_CODE_LENGTH = 10
- constexpr int32_t GPS_L5I_NH_CODE [10] = {0, 0, 0, 0, 1, 1, 0, 1, 0, 1}
- constexpr int32_t GPS_L5Q_NH_CODE_LENGTH = 20
- constexpr int32_t GPS_L5Q_NH_CODE [20] = {0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0}
- constexpr char GPS_L5I_NH_CODE_STR [11] = "0000110101"
- constexpr char GPS_L5Q_NH_CODE_STR [21] = "00000100110101001110"

13.774.1 Detailed Description

Defines system parameters for GPS L5 signal.

Author

Javier Arribas, 2017. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file GPS_L5.h.

13.775 GPS_L5.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file GPS_L5.h
00003  * \brief  Defines system parameters for GPS L5 signal
00004  * \author Javier Arribas, 2017. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_L5_H
00019 #define GNSS_SDR_GPS_L5_H
00020
00021 #include "GPS_CNAV.h"
00022 #include "gnss_frequencies.h"
00023 #include <cstdint>
00024
00025 /** \addtogroup Core
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 // carrier and code frequencies
00032 constexpr double GPS_L5_FREQ_HZ = FREQ5;              //!< L5 [Hz]
00033 constexpr double GPS_L5I_CODE_RATE_CPS = 10.23e6;     //!< GPS L5I code rate [chips/s]
00034 constexpr double GPS_L5I_PERIOD_S = 0.001;            //!< GPS L5I code period [seconds]
00035 constexpr double GPS_L5I_SYMBOL_PERIOD_S = 0.01;      //!< GPS L5I symbol period [seconds]
00036 constexpr double GPS_L5Q_CODE_RATE_CPS = 10.23e6;     //!< GPS L5Q code rate [chips/s]
00037 constexpr double GPS_L5Q_PERIOD_S = 0.001;            //!< GPS L5Q code period [seconds]
00038 constexpr int32_t GPS_L5Q_CODE_LENGTH_CHIPS = 10230;  //!< GPS L5Q code length [chips]
00039 constexpr int32_t GPS_L5I_CODE_LENGTH_CHIPS = 10230;  //!< GPS L5I code length [chips]
00040 constexpr int32_t GPS_L5I_PERIOD_MS = 1;              //!< GPS L5I code period [ms]
00041 constexpr int32_t GPS_L5I_SYMBOL_PERIOD_MS = 10;      //!< GPS L5I symbol period [ms]
00042
00043 constexpr int32_t GPS_L5_HISTORY_DEEP = 5;
00044
00045 // optimum parameters
00046 constexpr uint32_t GPS_L5_OPT_ACQ_FS_SPS = 10000000;  //!< Sampling frequency that maximizes the
      acquisition SNR while using a non-multiple of chip rate
00047
00048 constexpr int32_t GPS_L5I_INIT_REG[210] =
00049     {266, 365, 804, 1138,
00050         1509, 1559, 1756, 2084,
00051         2170, 2303, 2527, 2687,
00052         2930, 3471, 3940, 4132,
00053         4332, 4924, 5343, 5443,
00054         5641, 5816, 5898, 5918,
00055         5955, 6243, 6345, 6477,
00056         6518, 6875, 7168, 7187,
00057         7329, 7577, 7720, 7777,
00058         8057, 5358, 3550, 3412,
00059         819,
00060         4608, 3698, 962, 3001,
00061         4441, 4937, 3717, 4730,
00062         7291, 2279, 7613, 5723,
00063         7030, 1475, 2593, 2904,
00064         2056, 2757, 3756, 6205,
00065         5053, 6437,
00066         7789, 2311, 7432, 5155,
00067         1593, 5841, 5014, 1545,
00068         3016, 4875, 2119, 229,
00069         7634, 1406, 4506, 1819,
00070         7580, 5446, 6053, 7958,
00071         5267, 2956, 3544, 1277,
00072         2996, 1758, 3360, 2718,
00073         3754, 7440, 2781, 6756,
00074         7314, 208, 5252, 696,
00075         527, 1399, 5879, 6868,
00076         217, 7681, 3788, 1337,
00077         2424, 4243, 5686, 1955,
00078         4791, 492, 1518, 6566,
00079         5349, 506, 113, 1953,
00080         2797, 934, 3023, 3632,
00081         1330, 4909, 4867, 1183,
00082         3990, 6217, 1224, 1733,
```

```
00083         2319, 3928, 2380, 841,
00084         5049, 7027, 1197, 7208,
00085         8000, 152, 6762, 3745,
00086         4723, 5502, 4796, 123,
00087         8142, 5091, 7875, 330,
00088         5272, 4912, 374, 2045,
00089         6616, 6321, 7605, 2570,
00090         2419, 1234, 1922, 4317,
00091         5110, 825, 958, 1089,
00092         7813, 6058, 7703, 6702,
00093         1714, 6371, 2281, 1986,
00094         6282, 3201, 3760, 1056,
00095         6233, 1150, 2823, 6250,
00096         645, 2401, 1639, 2946,
00097         7091, 923, 7045, 6493,
00098         1706, 5836, 926, 6086,
00099         950, 5905, 3240, 6675,
00100         3197, 1555, 3589, 4555,
00101         5671, 6948, 4664, 2086,
00102         5950, 5521, 1515};
00103
00104
00105 constexpr int32_t GPS_L5Q_INIT_REG[210] =
00106     {
00107         1701, 323, 5292, 2020,
00108         5429, 7136, 1041, 5947,
00109         4315, 148, 535, 1939,
00110         5206, 5910, 3595, 5135,
00111         6082, 6990, 3546, 1523,
00112         4548, 4484, 1893, 3961,
00113         7106, 5299, 4660, 276,
00114         4389, 3783, 1591, 1601,
00115         749, 1387, 1661, 3210,
00116         708,
00117         4226, 5604, 6375, 3056,
00118         1772, 3662, 4401, 5218,
00119         2838, 6913, 1685, 1194,
00120         6963, 5001, 6694, 991,
00121         7489, 2441, 639, 2097,
00122         2498, 6470, 2399, 242,
00123         3768, 1186,
00124         5246, 4259, 5907, 3870,
00125         3262, 7387, 3069, 2999,
00126         7993, 7849, 4157, 5031,
00127         5986, 4833, 5739, 7846,
00128         898, 2022, 7446, 6404,
00129         155, 7862, 7795, 6121,
00130         4840, 6585, 429, 6020,
00131         200, 1664, 1499, 7298,
00132         1305, 7323, 7544, 4438,
00133         2485, 3387, 7319, 1853,
00134         5781, 1874, 7555, 2132,
00135         6441, 6722, 1192, 2588,
00136         2188, 297, 1540, 4138,
00137         5231, 4789, 659, 871,
00138         6837, 1393, 7383, 611,
00139         4920, 5416, 1611, 2474,
00140         118, 1382, 1092, 7950,
00141         7223, 1769, 4721, 1252,
00142         5147, 2165, 7897, 4054,
00143         3498, 6571, 2858, 8126,
00144         7017, 1901, 181, 1114,
00145         5195, 7479, 4186, 3904,
00146         7128, 1396, 4513, 5967,
00147         2580, 2575, 7961, 2598,
00148         4508, 2090, 3685, 7748,
00149         684, 913, 5558, 2894,
00150         5858, 6432, 3813, 3573,
00151         7523, 5280, 3376, 7424,
00152         2918, 5793, 1747, 7079,
00153         2921, 2490, 4119, 3373,
00154         977, 681, 4273, 5419,
00155         5626, 1266, 5804, 2414,
00156         6444, 4757, 427, 5452,
00157         5182, 6606, 6531, 4268,
00158         3115, 6835, 862, 4856,
00159         2765, 37, 1943, 7977,
00160         2512, 4451, 4071};
00161
00162 constexpr int32_t GPS_L5_CNAV_DATA_PAGE_BITS = 300;  //!< GPS L5 CNAV page length, including preamble
      and CRC [bits]
00163 constexpr int32_t GPS_L5_SYMBOLS_PER_BIT = 2;
00164 constexpr int32_t GPS_L5_SAMPLES_PER_SYMBOL = 10;
00165 constexpr int32_t GPS_L5_CNAV_DATA_PAGE_SYMBOLS = 600;
00166 constexpr int32_t GPS_L5_CNAV_DATA_PAGE_DURATION_S = 6;
00167 constexpr int32_t GPS_L5I_NH_CODE_LENGTH = 10;
00168 constexpr int32_t GPS_L5I_NH_CODE[10] = {0, 0, 0, 0, 1, 1, 0, 1, 0, 1};
```

```
00169 constexpr int32_t GPS_L5Q_NH_CODE_LENGTH = 20;
00170 constexpr int32_t GPS_L5Q_NH_CODE[20] = {0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0};
00171 constexpr char GPS_L5I_NH_CODE_STR[11] = "0000110101";
00172 constexpr char GPS_L5Q_NH_CODE_STR[21] = "00000100110101001110";
00173
00174
00175 /** \} */
00176 /** \} */
00177 #endif  // GNSS_SDR_GPS_L5_H
```

13.776 gps_navigation_message.h File Reference

Interface of a GPS NAV Data message decoder.

```
#include "GPS_L1_CA.h"
#include "gps_almanac.h"
#include "gps_ephemeris.h"
#include "gps_iono.h"
#include "gps_utc_model.h"
#include "qzss.h"
#include <bitset>
#include <cstdint>
#include <map>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class Gps_Navigation_Message

  This class decodes a GPS NAV Data message as described in IS-GPS-200M.

Enumerations

- enum class **LnavSystem** { **GPS** , **QZSS** }

13.776.1 Detailed Description

Interface of a GPS NAV Data message decoder.

Author

Javier Arribas, 2011. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file gps_navigation_message.h.

13.777 gps_navigation_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_navigation_message.h
00003  * \brief  Interface of a GPS NAV Data message decoder
00004  * \author Javier Arribas, 2011. jarribas(at)cttc.es
00005  *
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
```

```
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GPS_NAVIGATION_MESSAGE_H
00020 #define GNSS_SDR_GPS_NAVIGATION_MESSAGE_H
00021
00022
00023 #include "GPS_L1_CA.h"
00024 #include "gps_almanac.h"
00025 #include "gps_ephemeris.h"
00026 #include "gps_iono.h"
00027 #include "gps_utc_model.h"
00028 #include "qzss.h"
00029 #include <bitset>
00030 #include <cstdint>
00031 #include <map>
00032 #include <string>
00033 #include <utility>  // for pair
00034 #include <vector>
00035
00036 /** \addtogroup Core
00037  * \{ */
00038 /** \addtogroup System_Parameters
00039  * \{ */
00040
00041 enum class LnavSystem
00042 {
00043     GPS,
00044     QZSS
00045 };
00046
00047 /*!
00048  * \brief This class decodes a GPS NAV Data message as described in IS-GPS-200M
00049  *
00050  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
00051  */
00052 class Gps_Navigation_Message
00053 {
00054 public:
00055     /*!
00056      * Default constructor
00057      */
00058     explicit Gps_Navigation_Message(LnavSystem system = LnavSystem::GPS);
00059
00060     /*!
00061      * \brief Obtain a GPS SV Ephemeris class filled with current SV data
00062      */
00063     Gps_Ephemeris get_ephemeris() const;
00064
00065     /*!
00066      * \brief Obtain a GPS ionospheric correction parameters class filled with current SV data
00067      */
00068     Gps_Iono get_iono();
00069
00070     /*!
00071      * \brief Obtain a GPS almanac class filled with current SV data
00072      */
00073     Gps_Almanac get_almanac();
00074
00075     /*!
00076      * \brief Obtain a GPS Almanac  model parameters class filled with current SV data
00077      */
00078     Gps_Utc_Model get_utc_model();
00079
00080     /*!
00081      * \brief Decodes the GPS NAV message
00082      */
00083     int32_t subframe_decoder(const char* subframe);
00084
00085     /*!
00086      * \brief Computes the Coordinated Universal Time (UTC) and
00087      * returns it in [s] (IS-GPS-200M, 20.3.3.5.2.4)
00088      */
00089     double utc_time(double gpstime_corrected) const;
00090
00091     /*!
00092      * \brief Gets Time of Week, in seconds
00093      */
00094     inline int32_t get_TOW() const
00095     {
00096         return d_TOW;
00097     }
00098
00099     /*!
00100      * \brief Gets GPS Week Number
00101      */
00102     inline int32_t get_GPS_week() const
```

```
00103     {
00104         return i_GPS_week;
00105     }
00106
00107     /*!
00108      * \brief Sets satellite PRN number
00109      */
00110     inline void set_satellite_PRN(uint32_t prn)
00111     {
00112         i_satellite_PRN = prn;
00113     }
00114
00115     /*!
00116      * \brief Gets satellite PRN number
00117      */
00118     inline uint32_t get_satellite_PRN() const
00119     {
00120         return i_satellite_PRN;
00121     }
00122
00123     /*!
00124      * \brief Sets channel ID
00125      */
00126     inline void set_channel(int32_t channel_id)
00127     {
00128         i_channel_ID = channel_id;
00129     }
00130
00131     /*!
00132      * \brief Gets flag_iono_valid
00133      */
00134     inline bool get_flag_iono_valid() const
00135     {
00136         return flag_iono_valid;
00137     }
00138
00139     /*!
00140      * \brief Gets flag_utc_model_valid
00141      */
00142     inline bool get_flag_utc_model_valid() const
00143     {
00144         return flag_utc_model_valid;
00145     }
00146
00147     /*!
00148      * \brief Gets the almanac health field for a satellite PRN
00149      */
00150     int32_t get_almanac_health(uint32_t prn) const;
00151
00152     /*!
00153      * \brief Gets the 4-bit A-S flag and SV configuration term (page 25 of
00154      * subframe 4) of a satellite, or -1 if not decoded yet.
00155      */
00156     int32_t get_almanac_config_code(uint32_t prn) const;
00157
00158     bool satellite_validation();
00159     bool almanac_validation() const;
00160
00161 private:
00162     uint64_t read_navigation_unsigned(const std::bitset<GPS_SUBFRAME_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00163     int64_t read_navigation_signed(const std::bitset<GPS_SUBFRAME_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00164     bool read_navigation_bool(const std::bitset<GPS_SUBFRAME_BITS>& bits, const
      std::vector<std::pair<int32_t, int32_t»& parameter) const;
00165     void print_gps_word_bytes(uint32_t GPS_word) const;
00166     void decode_lnav_almanac(const std::bitset<GPS_SUBFRAME_BITS>& subframe_bits, uint32_t prn, double
      eccentricity_ref, double inclination_ref);
00167     void decode_lnav_iono_utc(const std::bitset<GPS_SUBFRAME_BITS>& subframe_bits);
00168     void decode_gps_almanac_health_sf4(const std::bitset<GPS_SUBFRAME_BITS>& subframe_bits);
00169     void decode_gps_almanac_health_sf5(const std::bitset<GPS_SUBFRAME_BITS>& subframe_bits);
00170     void decode_qzss_almanac_epoch_health(const std::bitset<GPS_SUBFRAME_BITS>& subframe_bits);
00171
00172     std::map<int32_t, int32_t> almanacHealth;      //!< Map that stores the health information stored
      in the almanac
00173     std::map<int32_t, int32_t> almanacConfigCode;  //!< A-S flag and SV configuration (page 25) per
      PRN
00174
00175     std::map<int32_t, std::string> satelliteBlock;  //!< Map that stores to which block the PRN
      belongs https://www.navcen.uscg.gov/?Do=constellationStatus
00176
00177     LnavSystem d_system;
00178
00179     // broadcast orbit 1
00180     int32_t d_TOW{};      // Time of GPS Week of the ephemeris set (taken from subframes TOW) [s]
00181     int32_t d_TOW_SF1{};  // Time of GPS Week from HOW word of Subframe 1 [s]
00182     int32_t d_TOW_SF2{};  // Time of GPS Week from HOW word of Subframe 2 [s]
```

```
00183     int32_t d_TOW_SF3{};  // Time of GPS Week from HOW word of Subframe 3 [s]
00184     int32_t d_TOW_SF4{};  // Time of GPS Week from HOW word of Subframe 4 [s]
00185     int32_t d_TOW_SF5{};  // Time of GPS Week from HOW word of Subframe 5 [s]
00186     int32_t d_IODE_SF2{};
00187     int32_t d_IODE_SF3{};
00188     double d_Crs{};      // Amplitude of the Sine Harmonic Correction Term to the Orbit Radius [m]
00189     double d_Delta_n{};  // Mean Motion Difference From Computed Value [semi-circles/s]
00190     double d_M_0{};      // Mean Anomaly at Reference Time [semi-circles]
00191     // broadcast orbit 2
00192     double d_Cuc{};             // Amplitude of the Cosine Harmonic Correction Term to the Argument of
      Latitude [rad]
00193     double d_e_eccentricity{};  // Eccentricity [dimensionless]
00194     double d_Cus{};             // Amplitude of the Sine Harmonic Correction Term to the Argument of
      Latitude [rad]
00195     double d_sqrt_A{};          // Square Root of the Semi-Major Axis [sqrt(m)]
00196     // broadcast orbit 3
00197     int32_t d_Toe{};    // Ephemeris data reference time of week (Ref. 20.3.3.4.3 IS-GPS-200M) [s]
00198     int32_t d_Toc{};    // clock data reference time (Ref. 20.3.3.3.3.1 IS-GPS-200M) [s]
00199     double d_Cic{};     // Amplitude of the Cosine Harmonic Correction Term to the Angle of
      Inclination [rad]
00200     double d_OMEGA0{};  // Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
00201     double d_Cis{};     // Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination
      [rad]
00202     // broadcast orbit 4
00203     double d_i_0{};        // Inclination Angle at Reference Time [semi-circles]
00204     double d_Crc{};        // Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius [m]
00205     double d_OMEGA{};      // Argument of Perigee [semi-cicles]
00206     double d_OMEGA_DOT{};  // Rate of Right Ascension [semi-circles/s]
00207     // broadcast orbit 5
00208     double d_IDOT{};          // Rate of Inclination Angle [semi-circles/s]
00209     int32_t i_code_on_L2{};   // If 1, P code ON in L2;  if 2, C/A code ON in L2;
00210     int32_t i_GPS_week{};     // GPS week number, aka WN [week]
00211     bool b_L2_P_data_flag{};  // When true, indicates that the NAV data stream was commanded OFF on
      the P-code of the L2 channel
00212     // broadcast orbit 6
00213     int32_t i_SV_accuracy{};  // User Range Accuracy (URA) index of the SV (reference paragraph 6.2.1)
      for the standard positioning service user (Ref 20.3.3.3.1.3 IS-GPS-200M)
00214     int32_t i_SV_health{};
00215     double d_TGD{};    // Estimated Group Delay Differential: L1-L2 correction term only for the
      benefit of "L1 P(Y)" or "L2 P(Y)" s users [s]
00216     int32_t d_IODC{};  // Issue of Data, Clock
00217     // broadcast orbit 7
00218     int32_t i_AODO{};            // Age of Data Offset (AODO) term for the navigation message
      correction table (NMCT) contained in subframe 4 (reference paragraph 20.3.3.5.1.9) [s]
00219     bool b_fit_interval_flag{};  // indicates the curve-fit interval used by the CS (Block
      II/IIA/IIR/IIR-M/IIF) and SS (Block IIIA) in determining the ephemeris parameters, as follows: 0 = 4
      hours, 1 = greater than 4 hours.
00220     double d_spare1{};
00221     double d_spare2{};
00222     double d_A_f0{};  // Coefficient 0 of code phase offset model [s]
00223     double d_A_f1{};  // Coefficient 1 of code phase offset model [s/s]
00224     double d_A_f2{};  // Coefficient 2 of code phase offset model [s/s^2]
00225
00226     // Almanac
00227     int32_t i_Toa{};      // Almanac reference time [s]
00228     int32_t i_WN_A{};     // Modulo 256 of the GPS week number to which the almanac reference time
      (i_Toa) is referenced
00229     int32_t SV_Health{};  // Almanac SV health
00230     uint32_t a_PRN{};     // Almanac PRN
00231     double a_delta_i{};   // Inclination Angle at Reference Time (relative to i_0 = 0.30 semi-circles)
00232     double a_M_0{};       // Mean Anomaly at Reference Time [semi-circles]
00233     double a_ecc{};       // Eccentricity [dimensionless]
00234     double a_sqrtA{};     // Square Root of the Semi-Major Axis [sqrt(m)]
00235     double a_OMEGA_0{};   // Longitude of Ascending Node of Orbit Plane at Weekly Epoch [semi-circles]
00236     double a_omega{};     // Argument of Perigee [semi-cicles]
00237     double a_OMEGAdot{};  // Rate of Right Ascension [semi-circles/s]
00238     double a_af0{};       // Coefficient 0 of code phase offset model [s]
00239     double a_af1{};       // Coefficient 1 of code phase offset model [s/s]
00240
00241     // satellite identification info
00242     int32_t i_channel_ID{};
00243     uint32_t i_satellite_PRN{};
00244
00245     // Ionospheric parameters
00246     double d_alpha0{};  // Coefficient 0 of a cubic equation representing the amplitude of the
      vertical delay [s]
00247     double d_alpha1{};  // Coefficient 1 of a cubic equation representing the amplitude of the
      vertical delay [s/semi-circle]
00248     double d_alpha2{};  // Coefficient 2 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^2]
00249     double d_alpha3{};  // Coefficient 3 of a cubic equation representing the amplitude of the
      vertical delay [s(semi-circle)^3]
00250     double d_beta0{};   // Coefficient 0 of a cubic equation representing the period of the model [s]
00251     double d_beta1{};   // Coefficient 1 of a cubic equation representing the period of the model
      [s/semi-circle]
00252     double d_beta2{};   // Coefficient 2 of a cubic equation representing the period of the model
      [s(semi-circle)^2]
```

```
00253     double d_beta3{};   // Coefficient 3 of a cubic equation representing the period of the model
      [s(semi-circle)^3]
00254
00255     // UTC parameters
00256     double d_A0{};  // Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4
      IS-GPS-200M) [s]
00257     double d_A1{};  // 1st order term of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4
      IS-GPS-200M) [s/s]
00258
00259     int32_t d_t_OT{};        // Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4
      IS-GPS-200M) [s]
00260     int32_t i_WN_T{};        // UTC reference week number [weeks]
00261     int32_t d_DeltaT_LS{};   // delta time due to leap seconds [s]. Number of leap seconds since
      6-Jan-1980 as transmitted by the GPS almanac.
00262     int32_t i_WN_LSF{};      // Week number at the end of which the leap second becomes effective
      [weeks]
00263     int32_t i_DN{};          // Day number (DN) at the end of which the leap second becomes effective
      [days]
00264     int32_t d_DeltaT_LSF{};  // Scheduled future or recent past (relative to NAV message upload) value
      of the delta time due to leap seconds [s]
00265
00266     // Flags
00267     bool b_valid_ephemeris_set_flag{};  // flag indicating that this ephemeris set have passed the
      validation check
00268     bool flag_iono_valid{};             // If set, it indicates that the ionospheric parameters are
      filled (page 18 has arrived and decoded)
00269     bool flag_utc_model_valid{};        // If set, it indicates that the UTC model parameters are
      filled
00270     bool flag_almanac_valid{};          // If set, it indicates that the almanac is filled
00271     bool flag_almanac_week_valid{};     // If set, it indicates that the almanac week is valid
00272     /*  If true, enhanced level of integrity assurance.
00273      *
00274      *  If false, indicates that the conveying signal is provided with the legacy level of integrity
      assurance.
00275      *  That is, the probability that the instantaneous URE of the conveying signal exceeds 4.42 times
      the upper bound
00276      *  value of the current broadcast URA index, for more than 5.2 seconds, without an accompanying
      alert, is less
00277      *  than 1E-5 per hour. If true, indicates that the conveying signal is provided with an enhanced
      level of
00278      *  integrity assurance. That is, the probability that the instantaneous URE of the conveying
      signal exceeds 5.73
00279      *  times the upper bound value of the current broadcast URA index, for more than 5.2 seconds,
      without an
00280      *  accompanying alert, is less than 1E-8 per hour.
00281      */
00282     bool b_integrity_status_flag{};
00283     bool b_alert_flag{};         // If true, indicates  that the SV URA may be worse than indicated in
      d_SV_accuracy, use that SV at our own risk.
00284     bool b_antispoofing_flag{};  // If true, the AntiSpoofing mode is ON in that SV
00285 };
00286
00287
00288 /** \} */
00289 /** \} */
00290 #endif  // GNSS_SDR_GPS_NAVIGATION_MESSAGE_H
```

13.778 gps_utc_model.h File Reference

Interface of a GPS UTC MODEL storage.

```
#include <boost/serialization/nvp.hpp>
#include <cstdint>
```

Classes

- class Gps_Utc_Model

  This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M.

13.778.1 Detailed Description

Interface of a GPS UTC MODEL storage.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_utc_model.h.

13.779 gps_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_utc_model.h
00003  * \brief  Interface of a GPS UTC MODEL storage
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_GPS_UTC_MODEL_H
00019 #define GNSS_SDR_GPS_UTC_MODEL_H
00020
00021 #include <boost/serialization/nvp.hpp>
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 /*!
00031  * \brief This class is a storage for the GPS UTC MODEL data as described in IS-GPS-200M
00032  *
00033  * See https://www.gps.gov/sites/default/files/2025-07/IS-GPS-200M.pdf Appendix II
00034  */
00035 class Gps_Utc_Model
00036 {
00037 public:
00038     /*!
00039      * Default constructor
00040      */
00041     Gps_Utc_Model() = default;
00042
00043     // UTC parameters
00044     double A0{};           //!< Constant of a model that relates GPS and UTC time (ref. 20.3.3.5.2.4
      IS-GPS-200M) [s]
00045     double A1{};           //!< 1st order term of a model that relates GPS and UTC time (ref.
      20.3.3.5.2.4 IS-GPS-200M) [s/s]
00046     double A2{};           //!< 2nd order term of a model that relates GPS and UTC time (ref.
      20.3.3.5.2.4 IS-GPS-200M) [s/s]
00047     int32_t tot{};         //!< Reference time for UTC data (reference 20.3.4.5 and 20.3.3.5.2.4
      IS-GPS-200M) [s]
00048     int32_t WN_T{};        //!< UTC reference week number [weeks]
00049     int32_t DeltaT_LS{};   //!< Delta time due to leap seconds [s]. Number of leap seconds since
      6-Jan-1980 as transmitted by the GPS almanac.
00050     int32_t WN_LSF{};      //!< Week number at the end of which the leap second becomes effective
      [weeks]
00051     int32_t DN{};          //!< Day number (DN) at the end of which the leap second becomes effective
      [days]
00052     int32_t DeltaT_LSF{};  //!< Scheduled future or recent past (relative to NAV message upload) value
      of the delta time due to leap seconds [s]
00053
00054     bool valid{};
00055
00056     template <class Archive>
00057     /*
00058      * \brief Serialize is a boost standard method to be called by the boost XML
00059      * serialization. Here is used to save the ephemeris data on disk file.
00060      */
00061     inline void serialize(Archive& archive, const uint32_t version)
00062     {
00063         if (version)
00064             {
```

```
00065             };
00066         archive& BOOST_SERIALIZATION_NVP(A0);
00067         archive& BOOST_SERIALIZATION_NVP(A1);
00068         archive& BOOST_SERIALIZATION_NVP(A2);
00069         archive& BOOST_SERIALIZATION_NVP(tot);
00070         archive& BOOST_SERIALIZATION_NVP(WN_T);
00071         archive& BOOST_SERIALIZATION_NVP(DeltaT_LS);
00072         archive& BOOST_SERIALIZATION_NVP(WN_LSF);
00073         archive& BOOST_SERIALIZATION_NVP(DN);
00074         archive& BOOST_SERIALIZATION_NVP(DeltaT_LSF);
00075         archive& BOOST_SERIALIZATION_NVP(valid);
00076     }
00077 };
00078
00079
00080 /** \} */
00081 /** \} */
00082 #endif  // GNSS_SDR_GPS_UTC_MODEL_H
```

13.780 gps_week_rollover.h File Reference

Resolution of the GPS mod-1024 week-number rollover from the receiver configuration.

```
#include <cstdint>
#include <string>
```

Functions

- int32_t gps_ref_week_from_config (const std::string &observation_date, bool pre_2009_file)

  Derives the reference GPS week number used to resolve the GPS mod-1024 week-number rollover when post-processing recorded signal files.

13.780.1 Detailed Description

Resolution of the GPS mod-1024 week-number rollover from the receiver configuration.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file gps_week_rollover.h.

13.781 gps_week_rollover.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file gps_week_rollover.h
00003  * \brief Resolution of the GPS mod-1024 week-number rollover from the
00004  * receiver configuration
00005  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_GPS_WEEK_ROLLOVER_H
00020 #define GNSS_SDR_GPS_WEEK_ROLLOVER_H
00021
00022 #include <cstdint>
00023 #include <string>
00024
00025 /** \addtogroup Core
```

```
00026  * \{ */
00027 /** \addtogroup System_Parameters
00028  * \{ */
00029
00030
00031 /*!
00032  * \brief Derives the reference GPS week number used to resolve the GPS
00033  * mod-1024 week-number rollover when post-processing recorded signal files.
00034  *
00035  * \param observation_date Value of the GNSS-SDR.observation_date
00036  * configuration option: approximate date of the signal capture, in
00037  * "YYYY-MM-DD" or "YYYY" format. An empty string means "not set".
00038  * \param pre_2009_file Value of the deprecated GNSS-SDR.pre_2009_file flag.
00039  *
00040  * \returns The full GPS week number of the observation date. Each mod-1024
00041  * broadcast week number is then resolved to the 1024-week era that places it
00042  * closest to this reference week (see adjgpsweek()). Returns 0 if neither
00043  * option is set, which keeps the default behavior of deriving the era from
00044  * the system clock. If only the deprecated flag is set, it returns week 1535
00045  * (mid-point of the 1999-2019 era), which reproduces the legacy fixed +1024
00046  * week adjustment, and a one-time deprecation notice is printed.
00047  */
00048 int32_t gps_ref_week_from_config(const std::string& observation_date, bool pre_2009_file);
00049
00050
00051 /** \} */
00052 /** \} */
00053 #endif  // GNSS_SDR_GPS_WEEK_ROLLOVER_H
```

13.782 MATH_CONSTANTS.h File Reference

Defines useful mathematical constants and their scaled versions.

Variables

- constexpr double GNSS_OMEGA_EARTH_DOT = 7.2921151467e-5

  Default Earth rotation rate, [rad/s].
- constexpr double SPEED_OF_LIGHT_M_S = 299792458.0

  Speed of light in vacuum [m/s].
- constexpr double SPEED_OF_LIGHT_M_MS = 299792.4580

  Speed of light in vacuum [m/ms].
- constexpr double GPS_GM = 3.986005e14

  Universal gravitational constant times the mass of the Earth, [m^3/s^2] IS-GPS-200M, 20.3.3.3.3.1.
- constexpr double GPS_F = -4.442807633e-10

  Constant, [s/(m)^(1/2)], IS-GPS-200M, 20.3.3.3.3.1.
- constexpr double GALILEO_GM = 3.986004418e14

  Geocentric gravitational constant[m^3/s^2], OS SIS ICD v2.0, pag. 44.
- constexpr double GALILEO_F = -4.442807309e-10

  Constant, [s/(m)^(1/2)]. OS SIS ICD v2.0, pag. 47.
- constexpr double GLONASS_OMEGA_EARTH_DOT = 7.292115e-5

  Earth rotation rate, [rad/s] ICD L1, L2 GLONASS Edition 5.1 2008 pag. 55.
- constexpr double GLONASS_GM = 398600.44e9

  Universal gravitational constant times the mass of the Earth, [m^3/s^2].
- constexpr double BEIDOU_OMEGA_EARTH_DOT = 7.2921150e-5

  Earth rotation rate, [rad/s] as defined in BDS-SIS-ICD-B1I-3.0 2019-02, pag. 3.
- constexpr double BEIDOU_GM = 3.986004418e14

  Universal gravitational constant times the mass of the Earth, [m^3/s^2] as defined in CGCS2000.
- constexpr double BEIDOU_F = -4.442807309e-10

  Constant, [s/(m)^(1/2)] F=-2(GM)^.5/C^2.
- constexpr double GNSS_PI = 3.1415926535898

  pi constant as defined for GNSS
- constexpr double HALF_PI = GNSS_PI / 2.0

pi/2

- constexpr double TWO_PI = 2.0 ∗ GNSS_PI

  2 ∗ pi
- constexpr double TWO_P3 = 8.0

  2^3
- constexpr double TWO_P4 = 16.0

  2^4
- constexpr double TWO_P8 = 256.0

  2^8
- constexpr double TWO_P11 = 2048.0

  2^11
- constexpr double TWO_P12 = 4096.0

  2^12
- constexpr double TWO_P14 = 16384.0

  2^14
- constexpr double TWO_P16 = 65536.0

  2^16
- constexpr double TWO_P19 = 524288.0

  2^19
- constexpr double TWO_P31 = 2147483648.0

  2^31
- constexpr double TWO_P32 = 4294967296.0

  2^32
- constexpr double TWO_P56 = 7.205759403792794e+016

  2^56
- constexpr double TWO_P57 = 1.441151880758559e+017

  2^57
- constexpr double TWO_N2 = 0.25

  2^-2
- constexpr double TWO_N5 = 0.03125

  2^-5
- constexpr double TWO_N6 = 0.015625

  2^-6
- constexpr double TWO_N8 = 0.00390625

  2^-8
- constexpr double TWO_N9 = 0.001953125

  2^-9
- constexpr double TWO_N10 = 0.0009765625

  2^-10
- constexpr double TWO_N11 = 4.882812500000000e-004

  2^-11
- constexpr double TWO_N14 = 0.00006103515625

  2^-14
- constexpr double TWO_N15 = 3.051757812500000e-005

  2^-15
- constexpr double TWO_N16 = 1.525878906250000e-005

  2^-16
- constexpr double TWO_N17 = 7.629394531250000e-006

  2^-17
- constexpr double TWO_N18 = 3.814697265625000e-006

  2^-18

- constexpr double TWO_N19 = 1.907348632812500e-006
  2^-19
- constexpr double TWO_N20 = 9.536743164062500e-007
  2^-20
- constexpr double TWO_N21 = 4.768371582031250e-007
  2^-21
- constexpr double TWO_N22 = 2.384185791015625e-007
  2^-22
- constexpr double TWO_N23 = 1.192092895507810e-007
  2^-23
- constexpr double TWO_N24 = 5.960464477539063e-008
  2^-24
- constexpr double TWO_N25 = 2.980232238769531e-008
  2^-25
- constexpr double TWO_N26 = 1.490116119384765e-009
  2^-26
- constexpr double TWO_N27 = 7.450580596923828e-009
  2^-27
- constexpr double TWO_N29 = 1.862645149230957e-009
  2^-29
- constexpr double TWO_N30 = 9.313225746154785e-010
  2^-30
- constexpr double TWO_N31 = 4.656612873077393e-010
  2^-31
- constexpr double TWO_N32 = 2.328306436538696e-010
  2^-32
- constexpr double TWO_N33 = 1.164153218269348e-010
  2^-33
- constexpr double TWO_N34 = 5.82076609134674e-011
  2^-34
- constexpr double TWO_N35 = 2.91038304567337e-011
  2^-35
- constexpr double TWO_N38 = 3.637978807091713e-012
  2^-38
- constexpr double TWO_N39 = 1.818989403545856e-012
  2^-39
- constexpr double TWO_N40 = 9.094947017729280e-013
  2^-40
- constexpr double TWO_N43 = 1.136868377216160e-013
  2^-43
- constexpr double TWO_N44 = 5.684341886080802e-14
  2^-44
- constexpr double TWO_N46 = 1.4210854715202e-014
  2^-46
- constexpr double TWO_N48 = 3.552713678800501e-15
  2^-46
- constexpr double TWO_N50 = 8.881784197001252e-016
  2^-50
- constexpr double TWO_N51 = 4.44089209850063e-016
  2^-51
- constexpr double TWO_N55 = 2.775557561562891e-017

2^-55

- constexpr double TWO_N57 = 6.938893903907228e-18

  2^-57
- constexpr double TWO_N59 = 1.73472347597681e-018

  2^-59
- constexpr double TWO_N60 = 8.673617379884036e-19

  2^-60
- constexpr double TWO_N66 = 1.3552527156068805425093160010874271392822265625e-20

  2^-66
- constexpr double TWO_N68 = 3.388131789017201e-21

  2^-68
- constexpr double PI_TWO_N19 = 5.992112452678286e-006

  *Pi*2^-19.*
- constexpr double PI_TWO_N43 = 3.571577341960839e-013

  *Pi*2^-43.*
- constexpr double PI_TWO_N31 = 1.462918079267160e-009

  *Pi*2^-31.*
- constexpr double PI_TWO_N38 = 1.142904749427469e-011

  *Pi*2^-38.*
- constexpr double PI_TWO_N23 = 3.745070282923929e-007

  *Pi*2^-23.*
- constexpr double D2R = GNSS_PI / 180.0

  deg to rad
- constexpr double R2D = 180.0 / GNSS_PI

  rad to deg
- constexpr double SC2RAD = GNSS_PI

  semi-circle to radian (IS-GPS)
- constexpr double AS2R = D2R / 3600.0

  arc sec to radian
- constexpr double AU = 149597870691.0

  1 Astronomical Unit AU (m) distance from Earth to the Sun.

13.782.1 Detailed Description

Defines useful mathematical constants and their scaled versions.

Author

Javier Arribas, 2013. jarribas(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file MATH_CONSTANTS.h.

13.783 MATH_CONSTANTS.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file MATH_CONSTANTS.h
00003  * \brief  Defines useful mathematical constants and their scaled versions
00004  * \author Javier Arribas, 2013. jarribas(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
```

```
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_MATH_CONSTANTS_H
00018 #define GNSS_SDR_MATH_CONSTANTS_H
00019
00020 /** \addtogroup Core
00021  * \{ */
00022 /** \addtogroup System_Parameters
00023  * \{ */
00024
00025
00026 constexpr double GNSS_OMEGA_EARTH_DOT = 7.2921151467e-5;  //!< Default Earth rotation rate, [rad/s]
00027 constexpr double SPEED_OF_LIGHT_M_S = 299792458.0;         //!< Speed of light in vacuum [m/s]
00028 constexpr double SPEED_OF_LIGHT_M_MS = 299792.4580;        //!< Speed of light in vacuum [m/ms]
00029
00030 // Physical constants for GPS
00031 constexpr double GPS_GM = 3.986005e14;       //!< Universal gravitational constant times the mass of
      the Earth, [m^3/s^2] IS-GPS-200M, 20.3.3.3.3.1
00032 constexpr double GPS_F = -4.442807633e-10;  //!< Constant, [s/(m)^(1/2)], IS-GPS-200M, 20.3.3.3.3.1
00033
00034 // Physical constants for Galileo
00035 constexpr double GALILEO_GM = 3.986004418e14;   //!< Geocentric gravitational constant[m^3/s^2], OS
      SIS ICD v2.0, pag. 44
00036 constexpr double GALILEO_F = -4.442807309e-10;  //!< Constant, [s/(m)^(1/2)]. OS SIS ICD v2.0, pag. 47
00037
00038 // Physical constants for GLONASS
00039 constexpr double GLONASS_OMEGA_EARTH_DOT = 7.292115e-5;  //!< Earth rotation rate, [rad/s] ICD L1, L2
      GLONASS Edition 5.1 2008 pag. 55
00040 constexpr double GLONASS_GM = 398600.44e9;               //!< Universal gravitational constant times
      the mass of the Earth, [m^3/s^2]
00041
00042 // Physical constants for Beidou
00043 constexpr double BEIDOU_OMEGA_EARTH_DOT = 7.2921150e-5;  //!< Earth rotation rate, [rad/s] as defined
      in BDS-SIS-ICD-B1I-3.0 2019-02, pag. 3
00044 constexpr double BEIDOU_GM = 3.986004418e14;             //!< Universal gravitational constant times
      the mass of the Earth, [m^3/s^2] as defined in CGCS2000
00045 constexpr double BEIDOU_F = -4.442807309e-10;           //!< Constant, [s/(m)^(1/2)] F=-2(GM)^.5/C^2
00046
00047 constexpr double GNSS_PI = 3.1415926535898;  //!< pi constant as defined for GNSS
00048 constexpr double HALF_PI = GNSS_PI / 2.0;    //!< pi/2
00049 constexpr double TWO_PI = 2.0 * GNSS_PI;     //!< 2 * pi
00050
00051
00052 // Constants for scaling the ephemeris found in the data message
00053 // the format is the following: TWO_N5 -> 2^-5, TWO_P4 -> 2^4, PI_TWO_N43 -> Pi*2^-43, etc etc
00054 // Additionally some of the PI*2^N terms are used in the tracking stuff
00055 //   TWO_PX ==> 2^X
00056 //   TWO_NX ==> 2^-X
00057 //   PI_TWO_NX ==> Pi*2^-X
00058
00059 constexpr double TWO_P3 = 8.0;                      //!< 2^3
00060 constexpr double TWO_P4 = 16.0;                     //!< 2^4
00061 constexpr double TWO_P8 = 256.0;                    //!< 2^8
00062 constexpr double TWO_P11 = 2048.0;                  //!< 2^11
00063 constexpr double TWO_P12 = 4096.0;                  //!< 2^12
00064 constexpr double TWO_P14 = 16384.0;                 //!< 2^14
00065 constexpr double TWO_P16 = 65536.0;                 //!< 2^16
00066 constexpr double TWO_P19 = 524288.0;                //!< 2^19
00067 constexpr double TWO_P31 = 2147483648.0;            //!< 2^31
00068 constexpr double TWO_P32 = 4294967296.0;            //!< 2^32
00069 constexpr double TWO_P56 = 7.205759403792794e+016;  //!< 2^56
00070 constexpr double TWO_P57 = 1.441151880758559e+017;  //!< 2^57
00071
00072 constexpr double TWO_N2 = 0.25;                     //!< 2^-2
00073 constexpr double TWO_N5 = 0.03125;                  //!< 2^-5
00074 constexpr double TWO_N6 = 0.015625;                 //!< 2^-6
00075 constexpr double TWO_N8 = 0.00390625;               //!< 2^-8
00076 constexpr double TWO_N9 = 0.001953125;              //!< 2^-9
00077 constexpr double TWO_N10 = 0.0009765625;            //!< 2^-10
00078 constexpr double TWO_N11 = 4.882812500000000e-004;  //!< 2^-11
00079 constexpr double TWO_N14 = 0.00006103515625;        //!< 2^-14
00080 constexpr double TWO_N15 = 3.051757812500000e-005;  //!< 2^-15
00081 constexpr double TWO_N16 = 1.525878906250000e-005;  //!< 2^-16
00082 constexpr double TWO_N17 = 7.629394531250000e-006;  //!< 2^-17
00083 constexpr double TWO_N18 = 3.814697265625000e-006;  //!< 2^-18
00084 constexpr double TWO_N19 = 1.907348632812500e-006;  //!< 2^-19
00085 constexpr double TWO_N20 = 9.536743164062500e-007;  //!< 2^-20
00086 constexpr double TWO_N21 = 4.768371582031250e-007;  //!< 2^-21
00087 constexpr double TWO_N22 = 2.384185791015625e-007;  //!< 2^-22
00088 constexpr double TWO_N23 = 1.192092895507810e-007;  //!< 2^-23
00089 constexpr double TWO_N24 = 5.960464477539063e-008;  //!< 2^-24
00090 constexpr double TWO_N25 = 2.980232238769531e-008;  //!< 2^-25
00091 constexpr double TWO_N26 = 1.490116119384765e-009;  //!< 2^-26
00092 constexpr double TWO_N27 = 7.450580596923828e-009;  //!< 2^-27
```

```
00093 constexpr double TWO_N29 = 1.862645149230957e-009;  //!< 2^-29
00094 constexpr double TWO_N30 = 9.313225746154785e-010;  //!< 2^-30
00095 constexpr double TWO_N31 = 4.656612873077393e-010;  //!< 2^-31
00096 constexpr double TWO_N32 = 2.328306436538696e-010;  //!< 2^-32
00097 constexpr double TWO_N33 = 1.164153218269348e-010;  //!< 2^-33
00098 constexpr double TWO_N34 = 5.82076609134674e-011;   //!< 2^-34
00099 constexpr double TWO_N35 = 2.91038304567337e-011;   //!< 2^-35
00100 constexpr double TWO_N38 = 3.637978807091713e-012;  //!< 2^-38
00101 constexpr double TWO_N39 = 1.818989403545856e-012;  //!< 2^-39
00102 constexpr double TWO_N40 = 9.094947017729280e-013;  //!< 2^-40
00103 constexpr double TWO_N43 = 1.136868377216160e-013;  //!< 2^-43
00104 constexpr double TWO_N44 = 5.684341886080802e-14;   //!< 2^-44
00105 constexpr double TWO_N46 = 1.4210854715202e-014;    //!< 2^-46
00106 constexpr double TWO_N48 = 3.552713678800501e-15;   //!< 2^-46
00107
00108 constexpr double TWO_N50 = 8.881784197001252e-016;                                  //!< 2^-50
00109 constexpr double TWO_N51 = 4.44089209850063e-016;                                   //!< 2^-51
00110 constexpr double TWO_N55 = 2.775557561562891e-017;                                  //!< 2^-55
00111 constexpr double TWO_N57 = 6.938893903907228e-18;                                   //!< 2^-57
00112 constexpr double TWO_N59 = 1.73472347597681e-018;                                   //!< 2^-59
00113 constexpr double TWO_N60 = 8.673617379884036e-19;                                   //!< 2^-60
00114 constexpr double TWO_N66 = 1.355252715606880542509316001087427139282226562 5e-20;  //!< 2^-66
00115 constexpr double TWO_N68 = 3.388131789017201e-21;                                   //!< 2^-68
00116
00117 constexpr double PI_TWO_N19 = 5.992112452678286e-006;  //!< Pi*2^-19
00118 constexpr double PI_TWO_N43 = 3.571577341960839e-013;  //!< Pi*2^-43
00119 constexpr double PI_TWO_N31 = 1.462918079267160e-009;  //!< Pi*2^-31
00120 constexpr double PI_TWO_N38 = 1.142904749427469e-011;  //!< Pi*2^-38
00121 constexpr double PI_TWO_N23 = 3.745070282923929e-007;  //!< Pi*2^-23
00122
00123 constexpr double D2R = GNSS_PI / 180.0;  //!< deg to rad
00124 constexpr double R2D = 180.0 / GNSS_PI;  //!< rad to deg
00125 constexpr double SC2RAD = GNSS_PI;       //!< semi-circle to radian (IS-GPS)
00126 constexpr double AS2R = D2R / 3600.0;    //!< arc sec to radian
00127
00128 constexpr double AU = 149597870691.0;  //!< 1 Astronomical Unit AU (m) distance from Earth to the Sun.
00129
00130
00131 /** \} */
00132 /** \} */
00133 #endif  // GNSS_SDR_MATH_CONSTANTS_H
```

13.784 osnma_data.h File Reference

Class for Galileo OSNMA data storage.

```
#include "galileo_inav_message.h"
#include <array>
#include <cstdint>
#include <memory>
#include <string>
#include <utility>
#include <vector>
```

Classes

- class DSM_nma_header
- class DSM_dsm_header
- class MACK_header
- class MACK_tag_info
- class MACK_tag_and_info
- class DSM_PKR_message
- class DSM_KROOT_message
- class MACK_message
- class OSNMA_NavData
- class OSNMA_data

  This class handles ONSMA data See https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf.
- class Tag

Functions

- uint64_t **osnma::galileo_gst_seconds** (uint32_t WN, uint32_t TOW)
- std::pair< uint32_t, uint32_t > **osnma::galileo_week_tow_with_offset** (uint32_t WN, uint32_t TOW, int32_t offset_seconds)
- uint32_t **osnma::galileo_week_to_uint** (int32_t WN)
- uint32_t **osnma::galileo_tow_to_uint** (int32_t TOW)
- bool **osnma::auth_gst_matches_nav_data** (uint64_t auth_gst, uint64_t nav_data_gst)
- bool **osnma::auth_gst_is_stale** (uint64_t auth_gst, uint64_t nav_data_gst)

Variables

- constexpr uint64_t osnma::GALILEO_SECONDS_PER_WEEK = 604800
- constexpr uint64_t osnma::OSNMA_AUTH_IOD_MAX_AGE_SECONDS = 4 * 60 * 60
- constexpr uint64_t osnma::OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS = 10 * 60

13.784.1 Detailed Description

Class for Galileo OSNMA data storage.

Author

Carles Fernandez-Prades, 2020-2023 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file osnma_data.h.

13.784.2 Variable Documentation

13.784.2.1 GALILEO_SECONDS_PER_WEEK

```
uint64_t osnma::GALILEO_SECONDS_PER_WEEK = 604800  [constexpr]
```

Definition at line 36 of file osnma_data.h.

13.784.2.2 OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS

```
uint64_t osnma::OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS = 10 * 60  [constexpr]
```

Definition at line 38 of file osnma_data.h.

13.784.2.3 OSNMA_AUTH_IOD_MAX_AGE_SECONDS

```
uint64_t osnma::OSNMA_AUTH_IOD_MAX_AGE_SECONDS = 4 * 60 * 60  [constexpr]
```

Definition at line 37 of file osnma_data.h.

13.785 osnma_data.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_data.h
00003  * \brief Class for Galileo OSNMA data storage
00004  * \author Carles Fernandez-Prades, 2020-2023 cfernandez(at)cttc.es
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_OSNMA_DATA_H
```

```
00019 #define GNSS_SDR_OSNMA_DATA_H
00020
00021 #include "galileo_inav_message.h"  // for OSNMA_msg
00022 #include <array>
00023 #include <cstdint>
00024 #include <memory>
00025 #include <string>
00026 #include <utility>
00027 #include <vector>
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup System_Parameters
00032  * \{ */
00033
00034 namespace osnma
00035 {
00036 constexpr uint64_t GALILEO_SECONDS_PER_WEEK = 604800;
00037 constexpr uint64_t OSNMA_AUTH_IOD_MAX_AGE_SECONDS = 4 * 60 * 60;
00038 constexpr uint64_t OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS = 10 * 60;
00039
00040 uint64_t galileo_gst_seconds(uint32_t WN, uint32_t TOW);
00041 std::pair<uint32_t, uint32_t> galileo_week_tow_with_offset(uint32_t WN, uint32_t TOW, int32_t
      offset_seconds);
00042 uint32_t galileo_week_to_uint(int32_t WN);
00043 uint32_t galileo_tow_to_uint(int32_t TOW);
00044 bool auth_gst_matches_nav_data(uint64_t auth_gst, uint64_t nav_data_gst);
00045 bool auth_gst_is_stale(uint64_t auth_gst, uint64_t nav_data_gst);
00046 }  // namespace osnma
00047
00048 class DSM_nma_header
00049 {
00050 public:
00051     DSM_nma_header() = default;
00052     uint8_t nmas{};
00053     uint8_t cid{};
00054     uint8_t cpks{};
00055     bool reserved{};
00056 };
00057
00058
00059 class DSM_dsm_header
00060 {
00061 public:
00062     DSM_dsm_header() = default;
00063     uint8_t dsm_id{};
00064     uint8_t dsm_block_id{};
00065 };
00066
00067
00068 class MACK_header
00069 {
00070 public:
00071     MACK_header() = default;
00072     uint64_t tag0{};
00073     uint16_t macseq{};
00074     uint8_t cop{};
00075     bool tag0_valid{false};
00076     bool macseq_valid{false};
00077 };
00078
00079
00080 class MACK_tag_info
00081 {
00082 public:
00083     MACK_tag_info() = default;
00084     uint8_t PRN_d{};
00085     uint8_t ADKD{};
00086     uint8_t cop{};
00087 };
00088
00089
00090 class MACK_tag_and_info
00091 {
00092 public:
00093     MACK_tag_and_info() = default;
00094     uint64_t tag{};  // C: 20-40 bits
00095     MACK_tag_info tag_info;
00096     uint32_t counter{};  // CTR
00097     bool valid{true};
00098 };
00099
00100
00101 class DSM_PKR_message
00102 {
00103 public:
00104     DSM_PKR_message() = default;
```

```
00105
00106     std::array<uint8_t, 128> itn{};  // bitset<1024>
00107     std::vector<uint8_t> npk;
00108     std::vector<uint8_t> p_dp;
00109     uint8_t nb_dp{};
00110     uint8_t mid{};
00111     uint8_t npkt{};
00112     uint8_t npktid{};
00113 };
00114
00115
00116 class DSM_KROOT_message
00117 {
00118 public:
00119     DSM_KROOT_message() = default;
00120
00121     std::vector<uint8_t> kroot;
00122     std::vector<uint8_t> ds;
00123     std::vector<uint8_t> p_dk;
00124     uint64_t alpha{};
00125     uint16_t wn_k{};
00126     uint8_t nb_dk{};
00127     uint8_t pkid{};
00128     uint8_t cidkr{};
00129     uint8_t reserved1{};
00130     uint8_t hf{};
00131     uint8_t mf{};
00132     uint8_t ks{};  // key size, in bits
00133     uint8_t ts{};
00134     uint8_t maclt{};
00135     uint8_t reserved{};
00136     uint8_t towh_k{};
00137     bool verified{false};
00138 };
00139
00140
00141 class MACK_message
00142 {
00143 public:
00144     MACK_message() = default;
00145     MACK_header header;
00146     std::vector<MACK_tag_and_info> tag_and_info;
00147     std::vector<uint8_t> key;
00148     std::vector<uint8_t> allowed_adkds{0, 4, 12};
00149     uint32_t TOW{};  // TODO duplicated variable, also in OSNMA_NavData
00150     uint32_t WN{};
00151     uint32_t PRNa{};
00152     uint8_t nmas{};
00153 };
00154
00155
00156 class OSNMA_NavData
00157 {
00158 public:
00159     OSNMA_NavData() : nav_data_id(id_counter++) {}
00160     const uint32_t nav_data_id;
00161     std::string get_utc_data() const;
00162     std::string get_ephemeris_data() const;
00163     uint32_t get_verified_bits() const { return verified_bits; }
00164     uint32_t get_prn_d() const { return PRNd; }
00165     uint32_t get_IOD_nav() const { return IOD_nav; }
00166     uint32_t get_wn_sf0() const { return d_WN_sf0; }
00167     uint32_t get_last_received_WN() const { return d_last_received_WN; }
00168     uint32_t get_last_received_TOW() const { return d_last_received_TOW; }
00169     uint32_t get_tow_sf0() const { return d_TOW_sf0; }
00170     uint64_t get_first_accumulated_tag_gst() const { return d_first_accumulated_tag_gst; }
00171     uint64_t get_last_accumulated_tag_gst() const { return d_last_accumulated_tag_gst; }
00172     uint8_t get_accumulated_tag_adkd() const { return d_accumulated_tag_adkd; }
00173     bool have_this_bits(std::string nav_data);
00174     bool get_verified_status() const { return verified; }
00175     bool has_tag_accumulation() const { return d_has_tag_accumulation; }
00176     bool add_nav_data(const std::string& nav_data);
00177     void set_tow_sf0(int value) { d_TOW_sf0 = value; }
00178     void set_wn_sf0(uint32_t WN) { d_WN_sf0 = WN; }
00179     void set_ephemeris_data(std::string value) { d_ephemeris_iono = value; }
00180     void set_utc_data(std::string value) { d_utc = value; }
00181     void update_last_received_timestamp(uint32_t TOW);
00182     void set_prn_d(uint32_t value) { PRNd = value; }
00183     void set_last_received_WN(uint32_t WN) { d_last_received_WN = WN; }
00184     void set_last_received_TOW(uint32_t TOW) { d_last_received_TOW = TOW; };
00185     void set_update_verified_bits(uint32_t morebits) { verified_bits += morebits; }
00186     void set_verified_status(bool value) { verified = value; }
00187     void set_IOD_nav(uint32_t value) { IOD_nav = value; }
00188     void start_tag_accumulation(uint32_t bits, uint8_t adkd, uint64_t tag_gst)
00189     {
00190         verified_bits = bits;
00191         d_accumulated_tag_adkd = adkd;
```

```
00192         d_first_accumulated_tag_gst = tag_gst;
00193         d_last_accumulated_tag_gst = tag_gst;
00194         d_has_tag_accumulation = true;
00195     }
00196     void continue_tag_accumulation(uint32_t morebits, uint64_t tag_gst)
00197     {
00198         verified_bits += morebits;
00199         d_last_accumulated_tag_gst = tag_gst;
00200     }
00201     void clear_tag_accumulation_metadata()
00202     {
00203         d_accumulated_tag_adkd = 0;
00204         d_first_accumulated_tag_gst = 0;
00205         d_last_accumulated_tag_gst = 0;
00206         d_has_tag_accumulation = false;
00207     }
00208     void reset_tag_accumulation()
00209     {
00210         verified_bits = 0;
00211         clear_tag_accumulation_metadata();
00212     }
00213
00214 private:
00215     static uint32_t id_counter;
00216     std::string d_ephemeris_iono{""};
00217     std::string d_utc{""};
00218     uint32_t d_WN_sf0{0};
00219     uint32_t d_last_received_WN{0};
00220     uint32_t d_TOW_sf0{0};
00221     uint32_t d_last_received_TOW{0};
00222     uint64_t d_first_accumulated_tag_gst{0};
00223     uint64_t d_last_accumulated_tag_gst{0};
00224     uint32_t PRNd{0};
00225     uint32_t verified_bits{0};
00226     uint32_t IOD_nav{0};
00227     uint8_t d_accumulated_tag_adkd{0};
00228     bool d_has_tag_accumulation{false};
00229     bool verified{false};
00230 };
00231
00232
00233 /*!
00234  * \brief This class handles ONSMA data
00235  * See
      https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_OSNMA_User_ICD_for_Test_Phase_v1.0.pdf
00236  */
00237 class OSNMA_data
00238 {
00239 public:
00240     OSNMA_data() = default;
00241     DSM_nma_header d_nma_header;
00242     DSM_dsm_header d_dsm_header;
00243     DSM_PKR_message d_dsm_pkr_message;
00244     DSM_KROOT_message d_dsm_kroot_message;
00245     DSM_KROOT_message d_dsm_kroot_new_message;
00246     MACK_message d_mack_message;
00247     OSNMA_NavData d_nav_data;
00248 };
00249
00250
00251 class Tag
00252 {
00253 public:
00254     enum e_verification_status
00255     {
00256         SUCCESS,
00257         FAIL,
00258         UNVERIFIED,
00259         AUTHENTICATED_DONT_USE
00260     };
00261     Tag(const MACK_tag_and_info& MTI,
00262         uint32_t TOW,
00263         uint32_t WN,
00264         uint32_t PRNa,
00265         uint8_t CTR,
00266         uint8_t nmas,
00267         std::vector<uint8_t> allowed_adkds = {0, 4, 12})  // standard tag constructor, for tags within
      Tag&Info field
00268         : tag_id(id_counter++),
00269           TOW(TOW),
00270           WN(WN),
00271           PRNa(PRNa),
00272           CTR(CTR),
00273           nmas(nmas),
00274           status(UNVERIFIED),
00275           received_tag(MTI.tag),
00276           computed_tag(0),
```

```
00277           PRN_d(MTI.tag_info.PRN_d),
00278           ADKD(MTI.tag_info.ADKD),
00279           cop(MTI.tag_info.cop),
00280           skipped(0),
00281           allowed_adkds(std::move(allowed_adkds))
00282     {
00283     }
00284     explicit Tag(const MACK_message& mack)  // constructor for Tag0
00285         : tag_id(id_counter++),
00286           TOW(mack.TOW),
00287           WN(mack.WN),
00288           PRNa(mack.PRNa),
00289           CTR(1),
00290           nmas(mack.nmas),
00291           status(UNVERIFIED),
00292           received_tag(mack.header.tag0),
00293           computed_tag(0),
00294           PRN_d(mack.PRNa),  // Tag0 are self-authenticating
00295           ADKD(0),
00296           cop(mack.header.cop),
00297           skipped(0),
00298           allowed_adkds(mack.allowed_adkds)
00299     {
00300     }
00301     const uint32_t tag_id;
00302     static uint32_t id_counter;
00303     uint32_t TOW;
00304     uint32_t WN;
00305     uint32_t PRNa;
00306     uint8_t CTR;
00307     uint8_t nmas;
00308     e_verification_status status;
00309     uint64_t received_tag;
00310     uint64_t computed_tag;
00311     uint8_t PRN_d;
00312     uint8_t ADKD;
00313     uint8_t cop;
00314     uint32_t skipped;
00315     std::vector<uint8_t> allowed_adkds{0, 4, 12};
00316     std::string nav_data;
00317 };
00318
00319 /** \} */
00320 /** \} */
00321
00322 #endif  // GNSS_SDR_OSNMA_DATA_H
```

13.786 osnma_dsm_reader.h File Reference

Class for reading OSNMA DSM messages.

```
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class OSNMA_DSM_Reader

13.786.1 Detailed Description

Class for reading OSNMA DSM messages.

Author

Carles Fernandez-Prades, 2023 cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2023 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file osnma_dsm_reader.h.

13.787 osnma_dsm_reader.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file osnma_dsm_reader.h
00003  * \brief Class for reading OSNMA DSM messages
00004  * \author Carles Fernandez-Prades, 2023 cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2023  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_OSNMA_DSM_READER_H
00018 #define GNSS_SDR_OSNMA_DSM_READER_H
00019
00020 #include <cstdint>
00021 #include <string>
00022 #include <vector>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029 class OSNMA_DSM_Reader
00030 {
00031 public:
00032     OSNMA_DSM_Reader() = default;
00033     uint8_t get_nmas(uint8_t nma_header) const;
00034     uint8_t get_cid(uint8_t nma_header) const;
00035     uint8_t get_cpks(uint8_t nma_header) const;
00036     bool get_nma_header_reserved(uint8_t nma_header) const;
00037
00038     uint8_t get_dsm_id(uint8_t dsm_header) const;
00039     uint8_t get_dsm_block_id(uint8_t dsm_header) const;
00040
00041     uint8_t get_number_blocks_index(uint8_t dsm_msg_0) const;
00042     uint8_t get_pkid(const std::vector<uint8_t>& dsm_msg) const;
00043     uint8_t get_cidkr(const std::vector<uint8_t>& dsm_msg) const;
00044     uint8_t get_dsm_reserved1(const std::vector<uint8_t>& dsm_msg) const;
00045     uint8_t get_hf(const std::vector<uint8_t>& dsm_msg) const;
00046     uint8_t get_mf(const std::vector<uint8_t>& dsm_msg) const;
00047     uint8_t get_ks(const std::vector<uint8_t>& dsm_msg) const;
00048     uint8_t get_ts(const std::vector<uint8_t>& dsm_msg) const;
00049     uint8_t get_maclt(const std::vector<uint8_t>& dsm_msg) const;
00050     uint8_t get_dsm_reserved(const std::vector<uint8_t>& dsm_msg) const;
00051     uint16_t get_wn_k(const std::vector<uint8_t>& dsm_msg) const;
00052     uint8_t get_towh_k(const std::vector<uint8_t>& dsm_msg) const;
00053     uint64_t get_alpha(const std::vector<uint8_t>& dsm_msg) const;
00054     uint16_t get_l_dk_bits(uint8_t nb_dk) const;
00055     uint16_t get_lk_bits(uint8_t ks) const;
00056     std::vector<uint8_t> get_kroot(const std::vector<uint8_t>& dsm_msg, uint16_t bytes_lk) const;
00057     std::string get_hash_function(uint8_t hf) const;
00058     std::string get_nmas_status(uint8_t nmas) const;
00059     std::string get_cpks_status(uint8_t cpks) const;
00060
00061     uint8_t get_mid(const std::vector<uint8_t>& dsm_msg) const;
00062     uint8_t get_npkt(const std::vector<uint8_t>& dsm_msg) const;
00063     uint8_t get_npktid(const std::vector<uint8_t>& dsm_msg) const;
00064
00065 private:
00066     static constexpr std::uint8_t mask_nmas{0xC0};
00067     static constexpr std::uint8_t mask_cid{0x30};
00068     static constexpr std::uint8_t mask_cpks{0x0E};
00069     static constexpr std::uint8_t mask_nma_header_reserved{0x01};
00070     static constexpr std::uint8_t mask_dsm_id{0xF0};
00071     static constexpr std::uint8_t mask_dsm_block_id{0x0F};
00072     static constexpr std::uint8_t mask_dsm_number_blocks{0xF0};
00073     static constexpr std::uint8_t mask_dsm_pkid{0x0F};
00074     static constexpr std::uint8_t mask_dsm_cidkr{0xC0};
00075     static constexpr std::uint8_t mask_dsm_reserved1{0x30};
00076     static constexpr std::uint8_t mask_dsm_hf{0x0C};
00077     static constexpr std::uint8_t mask_dsm_mf{0x03};
00078     static constexpr std::uint8_t mask_dsm_ks{0xF0};
00079     static constexpr std::uint8_t mask_dsm_ts{0x0F};
00080     static constexpr std::uint8_t mask_dsm_reserved{0xF0};
00081     static constexpr std::uint8_t mask_dsm_wk_k_msbyte{0x0F};
00082     static constexpr std::uint8_t mask_dsm_mid{0x0F};
00083     static constexpr std::uint8_t mask_dsm_npkt{0xF0};
00084     static constexpr std::uint8_t mask_dsm_npktid{0x0F};
00085 };
00086
00087 /** \} */
```

```
00088 /** \} */
00089 #endif  // GNSS_SDR_OSNMA_DSM_READER_H
```

13.788 qzss.h File Reference

Defines system parameters for QZSS signals.

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Functions

- constexpr uint32_t qzss_l1cb_prn_to_nominal_prn (uint32_t prn)

  Maps QZSS L1 C/B PRNs to the PRN carrying the satellite's nominal PNT signals. L1 C/B is broadcast with a dedicated PRN (203-206), but the transmitting satellite is identified by the PRN of its nominal signals (RINEX 4.00, Table 6: QZSS PRN to RINEX Satellite Identifier): 203->196 (J04), 204->197 (J05), 205->200 (J08), 206->201 (J09). Returns the input PRN unchanged if it is not an L1 C/B alias; in particular, PRN 202 (J10) broadcasts L1 C/B under its own PRN.

Variables

- constexpr const char QZSS_CA_PREAMBLE_SYMBOLS_STR [161] = "1111111111111111111100000000000000000000000
- constexpr const char QZSS_L5Q_NH_CODE_STR [21] = "00000100110101001110"
- constexpr const char QZSS_L5I_NH_CODE_STR [11] = "0000110101"

13.788.1 Detailed Description

Defines system parameters for QZSS signals.

Author

Carles Fernández-Prades, 2026. cfernandez (at) cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file qzss.h.

13.789 qzss.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss.h
00003  * \brief  Defines system parameters for QZSS signals
00004  * \author Carles Fernández-Prades, 2026. cfernandez (at) cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_QZSS_H
00019 #define GNSS_SDR_QZSS_H
00020
00021 #include "gnss_frequencies.h"
00022 #include <cstdint>
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
```

```
00029 static constexpr double QZSS_L1_FREQ_HZ = FREQ1;
00030 static constexpr double QZSS_L5_FREQ_HZ = FREQ5;
00031 static constexpr double QZSS_L1_CHIP_RATE = 1.023e6;
00032 static constexpr double QZSS_L5_CHIP_RATE = 10.23e6;
00033 static constexpr double QZSS_L1_CA_CODE_PERIOD_S = 0.001;
00034 static constexpr double QZSS_L5I_CODE_PERIOD_S = 0.001;
00035
00036 static constexpr int QZSS_L1_CODE_LENGTH = 1023;
00037 // The L1 replica is generated with 2 samples per chip, so a single buffer format
00038 // holds either the L1 C/A BPSK chips (each repeated twice) or the L1 C/B
00039 // sinBOC(1,1) subchips, selected per PRN (IS-QZSS-PNT-006)
00040 static constexpr int QZSS_L1_SAMPLES_PER_CHIP = 2;
00041 static constexpr int QZSS_L5_CODE_LENGTH = 10230;
00042 static constexpr int QZSS_L1_PERIOD_MS = 1;
00043 static constexpr int QZSS_L5I_PERIOD_MS = 1;
00044 static constexpr int32_t QZSS_CA_PREAMBLE_LENGTH_SYMBOLS = 160;
00045 static constexpr int32_t QZSS_CA_TELEMETRY_SYMBOLS_PER_BIT = 20;
00046 static constexpr int32_t QZSS_L5_SAMPLES_PER_SYMBOL = 10;
00047 static constexpr int32_t QZSS_L5Q_NH_CODE_LENGTH = 20;
00048 static constexpr int32_t QZSS_L5I_NH_CODE_LENGTH = 10;
00049 static constexpr uint32_t QZSS_L1_OPT_ACQ_FS_SPS = 2000000;
00050 static constexpr uint32_t QZSS_L5_OPT_ACQ_FS_SPS = 10000000;
00051
00052 static constexpr uint16_t XA_ALL_ONES = 0x1FFF;      // 13 bits all ones
00053 static constexpr uint16_t XA_SHORT_DECODE = 0x1FFD;  // 1111111111101 (ICD)
00054
00055 constexpr const char QZSS_CA_PREAMBLE_SYMBOLS_STR[161] =
      "11111111111111111111000000000000000000000000000000000000000000000000000000000000000011111111111111111111000000000000000000
00056 constexpr const char QZSS_L5Q_NH_CODE_STR[21] = "00000100110101001110";
00057 constexpr const char QZSS_L5I_NH_CODE_STR[11] = "0000110101";
00058
00059 static constexpr int32_t QZSS_LNAV_DATA_ID = 3;
00060 static constexpr int32_t QZSS_ALMANAC_EPOCH_HEALTH_SV_ID = 51;
00061 static constexpr int32_t QZSS_IONO_UTC_WIDE_AREA_SV_ID = 56;
00062 static constexpr int32_t QZSS_IONO_UTC_JAPAN_AREA_SV_ID = 61;
00063 static constexpr uint32_t QZSS_PRN_OFFSET = 192U;
00064 static constexpr uint32_t QZSS_L5_MAX_PRN = 202U;  //!< L5 PRN codes are only defined for PRNs 193-202
      (IS-QZSS-PNT, Table 3.2.5-1); L1 C/B PRNs 203-206 have no L5 counterpart
00065 static constexpr double QZSS_QZO_ECCENTRICITY_REF = 0.06;
00066 static constexpr double QZSS_QZO_INCLINATION_REF = 0.25;
00067
00068
00069 //! \brief Maps QZSS L1 C/B PRNs to the PRN carrying the satellite's nominal PNT signals.
00070 //! L1 C/B is broadcast with a dedicated PRN (203-206), but the transmitting satellite is
00071 //! identified by the PRN of its nominal signals (RINEX 4.00, Table 6: QZSS PRN to RINEX
00072 //! Satellite Identifier): 203->196 (J04), 204->197 (J05), 205->200 (J08), 206->201 (J09).
00073 //! Returns the input PRN unchanged if it is not an L1 C/B alias; in particular,
00074 //! PRN 202 (J10) broadcasts L1 C/B under its own PRN.
00075 constexpr uint32_t qzss_l1cb_prn_to_nominal_prn(uint32_t prn)
00076 {
00077     return prn == 203 ? 196 : (prn == 204 ? 197 : (prn == 205 ? 200 : (prn == 206 ? 201 : prn)));
00078 }
00079
00080 /** \} */
00081 /** \} */
00082
00083 #endif  // GNSS_SDR_QZSS_H
```

13.790 qzss_cnav_eop.h File Reference

Storage for QZSS CNAV Earth orientation parameters.

`#include "gps_cnav_eop.h"`

Classes

- class Qzss_CNAV_Eop

13.790.1 Detailed Description

Storage for QZSS CNAV Earth orientation parameters.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file qzss_cnav_eop.h.

13.791 qzss_cnav_eop.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_cnav_eop.h
00003  * \brief Storage for QZSS CNAV Earth orientation parameters
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * ----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * ----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_QZSS_CNAV_EOP_H
00019 #define GNSS_SDR_QZSS_CNAV_EOP_H
00020
00021
00022 #include "gps_cnav_eop.h"
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029
00030 class Qzss_CNAV_Eop : public Gps_CNAV_Eop
00031 {
00032 public:
00033     Qzss_CNAV_Eop() = default;
00034     explicit Qzss_CNAV_Eop(const Gps_CNAV_Eop& eop) : Gps_CNAV_Eop(eop) {}
00035 };
00036
00037
00038 /** \} */
00039 /** \} */
00040 #endif  // GNSS_SDR_QZSS_CNAV_EOP_H
```

13.792 qzss_cnav_iono.h File Reference

Interface of a QZSS CNAV ionospheric model storage.

`#include "gps_cnav_iono.h"`

Classes

- class Qzss_CNAV_Iono

  This class is a storage for the QZSS CNAV Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.

13.792.1 Detailed Description

Interface of a QZSS CNAV ionospheric model storage.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file qzss_cnav_iono.h.

13.793 qzss_cnav_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_cnav_iono.h
```

```
00003  * \brief Interface of a QZSS CNAV ionospheric model storage
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_QZSS_CNAV_IONO_H
00018 #define GNSS_SDR_QZSS_CNAV_IONO_H
00019
00020 #include "gps_cnav_iono.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027
00028 /*!
00029  * \brief This class is a storage for the QZSS CNAV Klobuchar ionospheric
00030  * model data as described in IS-QZSS-PNT-006
00031  *
00032  * Message Type 30 broadcasts the Wide Area coefficient set (the Japan area
00033  * set is broadcast in Message Type 61). See https://qzss.go.jp/en/technical/ps-is-qzss/
00034  */
00035 class Qzss_CNAV_Iono : public Gps_CNAV_Iono
00036 {
00037 public:
00038     Qzss_CNAV_Iono() = default;  //!< Default constructor
00039     explicit Qzss_CNAV_Iono(const Gps_CNAV_Iono& iono) : Gps_CNAV_Iono(iono) {}
00040 };
00041
00042
00043 /** \} */
00044 /** \} */
00045 #endif  // GNSS_SDR_QZSS_CNAV_IONO_H
```

13.794 qzss_cnav_utc_model.h File Reference

Interface of a QZSS CNAV UTC model storage.

`#include "gps_cnav_utc_model.h"`

Classes

- class Qzss_CNAV_Utc_Model

  This class is a storage for the QZSS CNAV UTC model data as described in IS-QZSS-PNT-006.

13.794.1 Detailed Description

Interface of a QZSS CNAV UTC model storage.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file qzss_cnav_utc_model.h.

13.795 qzss_cnav_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_cnav_utc_model.h
00003  * \brief Interface of a QZSS CNAV UTC model storage
```

```
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_QZSS_CNAV_UTC_MODEL_H
00018 #define GNSS_SDR_QZSS_CNAV_UTC_MODEL_H
00019
00020 #include "gps_cnav_utc_model.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027
00028 /*!
00029  * \brief This class is a storage for the QZSS CNAV UTC model data as
00030  * described in IS-QZSS-PNT-006
00031  *
00032  * The parameters share the GPS CNAV layout, but the broadcast offset refers
00033  * to UTC(NICT) instead of UTC(USNO). See https://qzss.go.jp/en/technical/ps-is-qzss/
00034  */
00035 class Qzss_CNAV_Utc_Model : public Gps_CNAV_Utc_Model
00036 {
00037 public:
00038     Qzss_CNAV_Utc_Model() = default;  //!< Default constructor
00039     explicit Qzss_CNAV_Utc_Model(const Gps_CNAV_Utc_Model& utc_model) : Gps_CNAV_Utc_Model(utc_model)
      {}
00040 };
00041
00042
00043 /** \} */
00044 /** \} */
00045 #endif  // GNSS_SDR_QZSS_CNAV_UTC_MODEL_H
```

13.796 qzss_iono.h File Reference

Interface of a QZSS ionospheric model storage.

`#include "gps_iono.h"`

Classes

- class Qzss_Iono

  This class is a storage for the QZSS Klobuchar ionospheric model data as described in IS-QZSS-PNT-006.

13.796.1 Detailed Description

Interface of a QZSS ionospheric model storage.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file qzss_iono.h.

13.797 qzss_iono.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_iono.h
00003  * \brief Interface of a QZSS ionospheric model storage
```

```
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_QZSS_IONO_H
00018 #define GNSS_SDR_QZSS_IONO_H
00019
00020 #include "gps_iono.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027
00028 /*!
00029  * \brief This class is a storage for the QZSS Klobuchar ionospheric model
00030  * data as described in IS-QZSS-PNT-006
00031  *
00032  * The coefficients share the GPS LNAV layout, but they are independently
00033  * determined by QZSS. See https://qzss.go.jp/en/technical/ps-is-qzss/
00034  */
00035 class Qzss_Iono : public Gps_Iono
00036 {
00037 public:
00038     Qzss_Iono() = default;  //!< Default constructor
00039     explicit Qzss_Iono(const Gps_Iono& iono) : Gps_Iono(iono) {}
00040 };
00041
00042
00043 /** \} */
00044 /** \} */
00045 #endif  // GNSS_SDR_QZSS_IONO_H
```

13.798 qzss_utc_model.h File Reference

Interface of a QZSS UTC model storage.
`#include "gps_utc_model.h"`

Classes

- class Qzss_Utc_Model

  This class is a storage for the QZSS UTC model data as described in IS-QZSS-PNT-006.

13.798.1 Detailed Description

Interface of a QZSS UTC model storage.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file qzss_utc_model.h.

13.799 qzss_utc_model.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file qzss_utc_model.h
00003  * \brief Interface of a QZSS UTC model storage
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
```

```
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_QZSS_UTC_MODEL_H
00018 #define GNSS_SDR_QZSS_UTC_MODEL_H
00019
00020 #include "gps_utc_model.h"
00021
00022 /** \addtogroup Core
00023  * \{ */
00024 /** \addtogroup System_Parameters
00025  * \{ */
00026
00027
00028 /*!
00029  * \brief This class is a storage for the QZSS UTC model data as described in
00030  * IS-QZSS-PNT-006
00031  *
00032  * The parameters share the GPS LNAV layout, but the broadcast offset refers
00033  * to UTC(NICT) instead of UTC(USNO). See https://qzss.go.jp/en/technical/ps-is-qzss/
00034  */
00035 class Qzss_Utc_Model : public Gps_Utc_Model
00036 {
00037 public:
00038     Qzss_Utc_Model() = default;  //!< Default constructor
00039     explicit Qzss_Utc_Model(const Gps_Utc_Model& utc_model) : Gps_Utc_Model(utc_model) {}
00040 };
00041
00042
00043 /** \} */
00044 /** \} */
00045 #endif  // GNSS_SDR_QZSS_UTC_MODEL_H
```

13.800 reed_solomon.h File Reference

Class implementing a Reed-Solomon encoder/decoder for blocks of 255 symbols and symbols of 8 bits.

```
#include <array>
#include <cstddef>
#include <cstdint>
#include <string>
#include <vector>
```

Classes

- class ReedSolomon

  Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where k=255-nroots is the information vector length and d=nroots+1 is the minimum Hamming distance, with symbols of 8 bits. It allows shortened RS codes.

13.800.1 Detailed Description

Class implementing a Reed-Solomon encoder/decoder for blocks of 255 symbols and symbols of 8 bits.

Author

Carles Fernandez, 2021. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2021 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file reed_solomon.h.

13.801 reed_solomon.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file reed_solomon.h
00003  * \brief Class implementing a Reed-Solomon encoder/decoder for blocks of 255
00004  * symbols and symbols of 8 bits.
00005  * \author Carles Fernandez, 2021. cfernandez(at)cttc.es
00006  *
00007  * -----------------------------------------------------------------------------
00008  *
00009  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00010  * This file is part of GNSS-SDR.
00011  *
00012  * Copyright (C) 2010-2021  (see AUTHORS file for a list of contributors)
00013  * SPDX-License-Identifier: GPL-3.0-or-later
00014  *
00015  * -----------------------------------------------------------------------------
00016  */
00017
00018
00019 #ifndef GNSS_SDR_REED_SOLOMON_H
00020 #define GNSS_SDR_REED_SOLOMON_H
00021
00022 #include <array>
00023 #include <cstddef>
00024 #include <cstdint>
00025 #include <string>
00026 #include <vector>
00027
00028
00029 /** \addtogroup Core
00030  * \{ */
00031 /** \addtogroup System_Parameters
00032  * \{ */
00033
00034 /*!
00035  * \brief
00036  * Class implementing a Reed-Solomon encoder and decoder RS(255,K,d) where
00037  * k=255-nroots is the information vector length and d=nroots+1 is the minimum
00038  * Hamming distance, with symbols of 8 bits. It allows shortened RS codes.
00039  */
00040 class ReedSolomon
00041 {
00042 public:
00043     /*!
00044      * \brief Default constructor.
00045      * Constructs a Reed Solomon object. The encode_with_generator_poly
00046      * and encode_with_generator_matrix methods are available for testing
00047      * purposes.
00048      *
00049      * gnss_signal: empty or "E6B" sets the Galileo E6B RS parameters.
00050      *  "E1B" sets the Galileo E1B (INAV) RS parameters.
00051      */
00052     explicit ReedSolomon(const std::string& gnss_signal = std::string());
00053
00054     /*!
00055      * \brief Custom constructor for RS(255, 255-nroots, nroots+1). Parameters:
00056      *
00057      * nroots - the number of roots in the RS code generator polynomial,
00058      *  which is the same as the number of parity symbols in a block.
00059      *
00060      * minpoly - primitive polynomial.
00061      *
00062      * prim - the primitive root of the generator polynomial.
00063      *
00064      * fcr - first consecutive root of the Reed-Solomon generator polynomial.
00065      *
00066      * pad - the number of pad symbols in a block. If not defined, it defaults
00067      *  to 0.
00068      *
00069      * shortening - value of the shortening parameter. Defaults to 0.
00070      *  If different to 0, it defines a shortened RS encoder/decoder.
00071      *
00072      * genpoly_coeff - a vector of (nroots+1) elements containing the generator
00073      *  polynomial coefficients. Only used for encoding. Defaults to empty.
00074      *  If defined, the encode_with_generator_poly method can be used.
00075      *
00076      * gen_matrix - a (255-shortening)x(255-nroots-shortening) matrix containing
00077      *  the elements of the generator matrix. Only used for encoding. Defaults
00078      *  to empty. If defined, the encode_with_generator_matrix method can be
00079      *  used.
00080      */
00081     ReedSolomon(int nroots,
00082         int minpoly,
00083         int prim,
00084         int fcr,
00085         int pad = 0,
00086         int shortening = 0,
00087         const std::vector<uint8_t>& genpoly_coeff = std::vector<uint8_t>{},
```

```
00088         const std::vector<std::vector<uint8_t»& gen_matrix = std::vector<std::vector<uint8_t»{});
00089
00090     /*!
00091      * \brief Decode an encoded block.
00092      *
00093      * The decoded symbols are at the first 255-nroots-shortening elements
00094      * of the data_to_decode vector.
00095      *
00096      * The second parameter is optional, and contains a vector of erasure
00097      * positions to be passed to the decoding algorithm. Defaults to empty.
00098      *
00099      * Returns the number of corrected errors, or -1 if decoding failed.
00100      */
00101     int decode(std::vector<uint8_t>& data_to_decode,
00102         const std::vector<int>& erasure_positions = std::vector<int>{}) const;
00103
00104     /*!
00105      * \brief Encode data with the generator matrix (for testing purposes)
00106      *
00107      * Returns the encoded vector. It is set to all zeros if the generator
00108      * matrix is not defined.
00109      */
00110     std::vector<uint8_t> encode_with_generator_matrix(const std::vector<uint8_t>& data_to_encode)
    const;
00111
00112     /*!
00113      * \brief Encode data with the generator polynomial (for testing purposes)
00114      *
00115      * Returns the encoded vector. It is set to all zeros if the generator
00116      * polynomial is not defined.
00117      */
00118     std::vector<uint8_t> encode_with_generator_poly(const std::vector<uint8_t>& data_to_encode) const;
00119
00120 private:
00121     static const int d_symbols_per_block = 255;  // the total number of symbols in a RS block.
00122     static const int d_symsize = 8;              // symbol size, in bits.
00123
00124     int mod255(int x) const;
00125     int rs_min(int a, int b) const;
00126     int decode_rs_8(uint8_t* data, const int* eras_pos, int no_eras) const;
00127
00128     uint8_t galois_mul(uint8_t a, uint8_t b) const;
00129     uint8_t galois_add(uint8_t a, uint8_t b) const;
00130     uint8_t galois_mul_table(uint8_t a, uint8_t b) const;
00131
00132     void encode_rs_8(const uint8_t* data, uint8_t* parity) const;
00133     void init_log_tables();    // initialize d_log_table and d_antilog
00134     void init_alpha_tables();  // initialize d_alpha_to, d_index_of
00135
00136     std::array<uint8_t, 256> d_alpha_to{};   // used for decoding
00137     std::array<uint8_t, 256> d_index_of{};   // used for decoding
00138     std::array<uint8_t, 256> d_log_table{};  // used for encoding
00139     std::array<uint8_t, 255> d_antilog{};    // used for encoding
00140
00141     std::vector<std::vector<uint8_t» d_genmatrix;  // used for encoding
00142     std::vector<uint8_t> d_genpoly_coeff;           // used for encoding
00143     std::vector<uint8_t> d_genpoly_index;           // used for encoding
00144
00145     size_t d_data_in_block{};           // number of information symbols in a block
00146     size_t d_rows_G{};                  // number of rows of the generator matrix
00147     size_t d_columns_G{};               // number of rows of the generator matrix
00148     size_t d_info_symbols_shortened{};  // number of info symbols in the shortened code
00149     size_t d_data_symbols_shortened{};  // number of data symbols in the shortened code
00150
00151     int d_nroots{};      // number of parity symbols in a block
00152     int d_prim{};        // The primitive root of the generator poly
00153     int d_pad{};         // the number of pad symbols in a block
00154     int d_iprim{};       // prim-th root of 1, index form
00155     int d_fcr{};         // first consecutive root
00156     int d_shortening{};  // shortening parameter
00157
00158     uint8_t d_min_poly{};  // primitive polynomial
00159     uint8_t d_a0{};        // auxiliary variable
00160 };
00161
00162 /** \} */
00163 /** \} */
00164 #endif  // GNSS_SDR_REED_SOLOMON_H
```

13.802 sbas_ephemeris.h File Reference

Interface of a SBAS REFERENCE LOCATION storage.

```
#include <ostream>
```

Classes

- class Sbas_Ephemeris

  This class stores SBAS SV ephemeris data.

13.802.1 Detailed Description

Interface of a SBAS REFERENCE LOCATION storage.

Author

Daniel Fehr, 2013. daniel.co(at)bluewin.ch

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2020 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sbas_ephemeris.h.

13.803 sbas_ephemeris.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sbas_ephemeris.h
00003  * \brief  Interface of a SBAS REFERENCE LOCATION storage
00004  * \author Daniel Fehr, 2013. daniel.co(at)bluewin.ch
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2020  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017
00018 #ifndef GNSS_SDR_SBAS_EPHEMERIS_H
00019 #define GNSS_SDR_SBAS_EPHEMERIS_H
00020
00021 #include <ostream>
00022
00023 /** \addtogroup Core
00024  * \{ */
00025 /** \addtogroup System_Parameters
00026  * \{ */
00027
00028
00029 /*!
00030  * \brief This class stores SBAS SV ephemeris data
00031  *
00032  */
00033 class Sbas_Ephemeris
00034 {
00035 public:
00036     Sbas_Ephemeris() = default;
00037
00038     int i_prn{};              //!< PRN number
00039     int i_t0{};               //!< Reference epoch time (GPST)
00040     double d_tof{};           //!< Time of message frame (GPST)
00041     int i_sv_ura{};           //!< SV accuracy (URA index), not standardized
00042     bool b_sv_do_not_use{};   //!< Health status (false:do not use / true:usable)
00043     double d_pos[3]{};        //!< Satellite position (m) (ECEF)
00044     double d_vel[3]{};        //!< Satellite velocity (m/s) (ECEF)
00045     double d_acc[3]{};        //!< Satellite acceleration (m/s^2) (ECEF)
00046     double d_af0{};           //!< Satellite clock-offset (s)
00047     double d_af1{};           //!< Satellite drift (s/s)
00048
00049     void print(std::ostream &out);
00050 };
00051
00052
00053 /** \} */
00054 /** \} */
00055 #endif  // GNSS_SDR_SBAS_EPHEMERIS_H
```

13.804 SBAS_L1.h File Reference

Defines system parameters for SBAS L1 signal.

```
#include "gnss_frequencies.h"
#include <cstdint>
```

Variables

- constexpr double SBAS_L1_FREQ_HZ = FREQ1

  SBAS L1 carrier frequency [Hz] (same as GPS L1).
- constexpr double SBAS_L1_CODE_RATE_CPS = 1.023e6

  SBAS L1 C/A code chip rate [chips/s].
- constexpr double SBAS_L1_CODE_LENGTH_CHIPS = 1023.0

  SBAS L1 C/A code length [chips].
- constexpr double SBAS_L1_CODE_PERIOD_S = 0.001

  SBAS L1 C/A code period [s].
- constexpr uint32_t SBAS_L1_CODE_PERIOD_MS = 1U

  SBAS L1 C/A code period [ms].
- constexpr uint32_t SBAS_L1_OPT_ACQ_FS_SPS = 2000000U

  Sampling frequency that maximises acquisition SNR [samples/s].
- constexpr uint32_t SBAS_L1_PRN_MIN = 120U

  Minimum SBAS PRN (inclusive).
- constexpr uint32_t SBAS_L1_PRN_MAX = 138U

  Maximum SBAS PRN (inclusive).
- constexpr int32_t SBAS_L1_DATA_RATE_BPS = 250

  Raw data bit rate [bits/s].
- constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_RATE_SPS = 500

  Encoded channel symbol rate (rate-1/2 FEC) [symbols/s].
- constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS = 2

  Encoded channel symbol period [ms] (1/500 sps).
- constexpr int32_t SBAS_L1_MSG_LENGTH_BITS = 250

  Information bits per 1-second message frame [bits].
- constexpr int32_t SBAS_L1_MSG_LENGTH_BYTES = 32

  Bytes holding a 250-bit frame with six trailing zero bits.
- constexpr int32_t SBAS_L1_DATA_FIELD_BYTES = 29

  Bytes holding preamble, type, and 212-bit data field.
- constexpr int32_t SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS = 1
- constexpr uint8_t SBAS_L1_PREAMBLE_1 = 0x53U

  First preamble byte (binary: 0101 0011).
- constexpr uint8_t SBAS_L1_PREAMBLE_2 = 0x9AU

  Second preamble byte (binary: 1001 1010).
- constexpr uint8_t SBAS_L1_PREAMBLE_3 = 0xC6U

  Third preamble byte (binary: 1100 0110).

13.804.1 Detailed Description

Defines system parameters for SBAS L1 signal.

Author

Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es

Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com

Reference: RTCA DO-229E, Minimum Operational Performance Standards for Global Positioning System/Wide Area Augmentation System Airborne Equipment

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file SBAS_L1.h.

13.805 SBAS_L1.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file SBAS_L1.h
00003  * \brief  Defines system parameters for SBAS L1 signal
00004  * \author Miguel Gómez López, 2026. mgomezl(at)ing.uc3m.es
00005  * \author Víctor Castillo Agüero, 2026. victorcastilloaguero(at)gmail.com
00006  *
00007  * Reference: RTCA DO-229E, Minimum Operational Performance Standards for
00008  * Global Positioning System/Wide Area Augmentation System Airborne Equipment
00009  *
00010  * -----------------------------------------------------------------------------
00011  *
00012  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00013  * This file is part of GNSS-SDR.
00014  *
00015  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00016  * SPDX-License-Identifier: GPL-3.0-or-later
00017  *
00018  * -----------------------------------------------------------------------------
00019  */
00020
00021
00022 #ifndef GNSS_SDR_SBAS_L1_H
00023 #define GNSS_SDR_SBAS_L1_H
00024
00025 #include "gnss_frequencies.h"
00026 #include <cstdint>
00027
00028 /** \addtogroup Core
00029  * \{ */
00030 /** \addtogroup System_Parameters
00031  * \{ */
00032
00033 // Carrier and code frequencies
00034 constexpr double SBAS_L1_FREQ_HZ = FREQ1;             //!< SBAS L1 carrier frequency [Hz] (same as GPS
      L1)
00035 constexpr double SBAS_L1_CODE_RATE_CPS = 1.023e6;     //!< SBAS L1 C/A code chip rate [chips/s]
00036 constexpr double SBAS_L1_CODE_LENGTH_CHIPS = 1023.0;  //!< SBAS L1 C/A code length [chips]
00037 constexpr double SBAS_L1_CODE_PERIOD_S = 0.001;       //!< SBAS L1 C/A code period [s]
00038 constexpr uint32_t SBAS_L1_CODE_PERIOD_MS = 1U;       //!< SBAS L1 C/A code period [ms]
00039
00040 // Optimum acquisition sampling frequency
00041 constexpr uint32_t SBAS_L1_OPT_ACQ_FS_SPS = 2000000U;  //!< Sampling frequency that maximises
      acquisition SNR [samples/s]
00042
00043 // PRN range (RTCA DO-229, Table A-1)
00044 constexpr uint32_t SBAS_L1_PRN_MIN = 120U;  //!< Minimum SBAS PRN (inclusive)
00045 constexpr uint32_t SBAS_L1_PRN_MAX = 138U;  //!< Maximum SBAS PRN (inclusive)
00046
00047 // Message rates and encoding
00048 //   SBAS broadcasts at 250 raw data bits/s, convolutionally encoded
00049 //   at rate 1/2, K=7, yielding a channel symbol rate of 500 symbols/s.
00050 constexpr int32_t SBAS_L1_DATA_RATE_BPS = 250;            //!< Raw data bit rate [bits/s]
00051 constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_RATE_SPS = 500;  //!< Encoded channel symbol rate (rate-1/2
      FEC) [symbols/s]
00052 constexpr int32_t SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS = 2;   //!< Encoded channel symbol period [ms]
      (1/500 sps)
00053 constexpr int32_t SBAS_L1_MSG_LENGTH_BITS = 250;          //!< Information bits per 1-second message
      frame [bits]
00054 constexpr int32_t SBAS_L1_MSG_LENGTH_BYTES = 32;          //!< Bytes holding a 250-bit frame with six
      trailing zero bits
00055 constexpr int32_t SBAS_L1_DATA_FIELD_BYTES = 29;          //!< Bytes holding preamble, type, and
      212-bit data field
00056
00057 // Maximum coherent integration extension in terms of 1 ms code periods.
00058 // Kept at 1 (no extension): the tracking correlator dumps are not
00059 // synchronized to the encoded symbol boundary, so extending beyond a single
00060 // code period would straddle that boundary and corrupt the sample pairing
```

```
00061 // that sbas_l1_telemetry_decoder_gs performs on its own.
00062 constexpr int32_t SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS = 1;
00063
00064 // Preamble bytes (RTCA DO-229, Section A.4.4.2)
00065 constexpr uint8_t SBAS_L1_PREAMBLE_1 = 0x53U;  //!< First preamble byte  (binary: 0101 0011)
00066 constexpr uint8_t SBAS_L1_PREAMBLE_2 = 0x9AU;  //!< Second preamble byte (binary: 1001 1010)
00067 constexpr uint8_t SBAS_L1_PREAMBLE_3 = 0xC6U;  //!< Third preamble byte  (binary: 1100 0110)
00068
00069 /** \} */
00070 /** \} */
00071
00072 #endif  // GNSS_SDR_SBAS_L1_H
```

13.806 sbas_raw_message.h File Reference

Container for a CRC-validated raw SBAS L1 message.

```
#include "SBAS_L1.h"
#include <algorithm>
#include <array>
#include <cstddef>
#include <cstdint>
#include <vector>
```

Classes

- class Sbas_Raw_Message

  A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp.

13.806.1 Detailed Description

Container for a CRC-validated raw SBAS L1 message.

Author

Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file sbas_raw_message.h.

13.807 sbas_raw_message.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file sbas_raw_message.h
00003  * \brief Container for a CRC-validated raw SBAS L1 message
00004  * \author Carles Fernandez-Prades, 2026. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SBAS_RAW_MESSAGE_H
00018 #define GNSS_SDR_SBAS_RAW_MESSAGE_H
00019
00020 #include "SBAS_L1.h"
00021 #include <algorithm>
00022 #include <array>
00023 #include <cstddef>
00024 #include <cstdint>
00025 #include <vector>
00026
00027 /** \addtogroup Core
```

```
00028  * \{ */
00029 /** \addtogroup System_Parameters
00030  * \{ */
00031
00032 /*!
00033  * \brief A CRC-validated 250-bit SBAS L1 frame and its receiver timestamp.
00034  *
00035  * The frame is stored MSB first as 32 bytes. Its six least-significant padding
00036  * bits are zero. The timestamp identifies the first bit of the preamble in the
00037  * receiver sample-clock time base.
00038  */
00039 class Sbas_Raw_Message
00040 {
00041 public:
00042     Sbas_Raw_Message() = default;
00043
00044     Sbas_Raw_Message(double sample_stamp_s, uint32_t prn, const std::vector<uint8_t>& frame)
00045         : d_sample_stamp_s(sample_stamp_s),
00046           d_prn(prn)
00047     {
00048         const size_t bytes_to_copy = std::min(frame.size(), d_frame.size());
00049         std::copy_n(frame.cbegin(), bytes_to_copy, d_frame.begin());
00050         d_frame.back() &= 0xC0U;
00051     }
00052
00053     double sample_stamp_s() const { return d_sample_stamp_s; }
00054     uint32_t prn() const { return d_prn; }
00055     uint32_t message_type() const { return (static_cast<uint32_t>(d_frame[1]) » 2U) & 0x3FU; }
00056     const std::array<uint8_t, SBAS_L1_MSG_LENGTH_BYTES>& frame() const { return d_frame; }
00057
00058 private:
00059     double d_sample_stamp_s{};
00060     uint32_t d_prn{};
00061     std::array<uint8_t, SBAS_L1_MSG_LENGTH_BYTES> d_frame{};
00062 };
00063
00064 /** \} */
00065 /** \} */
00066
00067 #endif  // GNSS_SDR_SBAS_RAW_MESSAGE_H
```

13.808 signal_flag.h File Reference

Enum for signals.

`#include <cstdint>`

Enumerations

- enum **signal_flag** : uint32_t {
  GPS_1C = 0x1 << 0 , **GPS_2S** = 0x1 << 1 , **GPS_L5** = 0x1 << 2 , **GAL_1B** = 0x1 << 3 ,
  GAL_E5a = 0x1 << 4 , **GAL_E5b** = 0x1 << 5 , **GAL_E6** = 0x1 << 6 , **GLO_1G** = 0x1 << 7 ,
  GLO_2G = 0x1 << 8 , **BDS_B1** = 0x1 << 9 , **BDS_B3** = 0x1 << 10 , **QZS_J1** = 0x1 << 11 ,
  QZS_J5 = 0x1 << 12 , **BDS_B1C** = 0x1 << 13 , **SBAS_S1** = 0x1 << 14 , **BDS_B2A** = 0x1 << 15 }

13.808.1 Detailed Description

Enum for signals.

Author

Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.com

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR.
Copyright (C) 2010-2026 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later
Definition in file signal_flag.h.

13.808.2 Enumeration Type Documentation

13.808.2.1 signal_flag

`enum signal_flag : uint32_t`

Definition at line 24 of file signal_flag.h.

13.809 signal_flag.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file signal_flag.h
00003  * \brief Enum for signals
00004  * \author Mathieu Favreau, 2026. favreau.mathieu(at)hotmail.com
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2026  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_SIGNAL_FLAG_H
00018 #define GNSS_SDR_SIGNAL_FLAG_H
00019
00020 #include <cstdint>
00021
00022 class ConfigurationInterface;
00023
00024 enum signal_flag : uint32_t
00025 {
00026     GPS_1C = 0x1 << 0,
00027     GPS_2S = 0x1 << 1,
00028     GPS_L5 = 0x1 << 2,
00029     GAL_1B = 0x1 << 3,
00030     GAL_E5a = 0x1 << 4,
00031     GAL_E5b = 0x1 << 5,
00032     GAL_E6 = 0x1 << 6,
00033     GLO_1G = 0x1 << 7,
00034     GLO_2G = 0x1 << 8,
00035     BDS_B1 = 0x1 << 9,
00036     BDS_B3 = 0x1 << 10,
00037     QZS_J1 = 0x1 << 11,
00038     QZS_J5 = 0x1 << 12,
00039     BDS_B1C = 0x1 << 13,
00040     SBAS_S1 = 0x1 << 14,
00041     BDS_B2A = 0x1 << 15
00042 };
00043
00044 #endif  // GNSS_SDR_SIGNAL_FLAG_H
```

13.810 tow_to_trk.h File Reference

Class to inform about TOW from Telemetry to Tracking blocks.

```
#include <cstdint>
#include <string>
```

Classes

- class TOW_to_trk

13.810.1 Detailed Description

Class to inform about TOW from Telemetry to Tracking blocks.

Author

Carles Fernandez, 2025. cfernandez(at)cttc.es

GNSS-SDR is a Global Navigation Satellite System software-defined receiver. This file is part of GNSS-SDR. Copyright (C) 2010-2025 (see AUTHORS file for a list of contributors) SPDX-License-Identifier: GPL-3.0-or-later

Definition in file tow_to_trk.h.

13.811 tow_to_trk.h

Go to the documentation of this file.

```
00001 /*!
00002  * \file tow_to_trk.h
00003  * \brief Class to inform about TOW from Telemetry to Tracking blocks
00004  * \author Carles Fernandez, 2025. cfernandez(at)cttc.es
00005  *
00006  * -----------------------------------------------------------------------------
00007  *
00008  * GNSS-SDR is a Global Navigation Satellite System software-defined receiver.
00009  * This file is part of GNSS-SDR.
00010  *
00011  * Copyright (C) 2010-2025  (see AUTHORS file for a list of contributors)
00012  * SPDX-License-Identifier: GPL-3.0-or-later
00013  *
00014  * -----------------------------------------------------------------------------
00015  */
00016
00017 #ifndef GNSS_SDR_TOW_TO_TRK_H
00018 #define GNSS_SDR_TOW_TO_TRK_H
00019
00020 #include <cstdint>
00021 #include <string>
00022
00023
00024 /** \addtogroup Core
00025  * \{ */
00026 /** \addtogroup System_Parameters
00027  * \{ */
00028
00029 class TOW_to_trk
00030 {
00031 public:
00032     TOW_to_trk() = default;
00033
00034     // Constructor with all parameters
00035     TOW_to_trk(const std::string& sig, int32_t ch, uint32_t t, uint64_t stamp, int32_t w, uint32_t p)
00036         : signal(sig), channel(ch), tow(t), sample_stamp(stamp), wn(w), prn(p) {}
00037
00038     std::string signal;
00039     int32_t channel{0};
00040     uint32_t tow{0};
00041     uint64_t sample_stamp{0};
00042     int32_t wn{0};
00043     uint32_t prn{0};
00044 };
00045
00046 /** \} */
00047 /** \} */
00048 #endif  // GNSS_SDR_TOW_TO_TRK_H
```

Index

~An_Packet_Printer
An_Packet_Printer, 355
~ArraySignalConditioner
ArraySignalConditioner, 357
~Channel
Channel, 404
~ControlThread
ControlThread, 428
~DllPllTrackingAdapter
DllPllTrackingAdapter, 461
~Exponential_Smoother
Exponential_Smoother, 479
~FirFilter
FirFilter, 498
~Fpga_Acquisition
Fpga_Acquisition, 508
~Fpga_DMA
Fpga_DMA, 511
~Fpga_Multicorrelator_8sc
Fpga_Multicorrelator_8sc, 514
~Fpga_Switch
Fpga_Switch, 517
~Fpga_buffer_monitor
Fpga_buffer_monitor, 510
~Fpga_dynamic_bit_selection
Fpga_dynamic_bit_selection, 512
~Fpga_spidev
Fpga_spidev, 516
~GNSSFlowgraph
GNSSFlowgraph, 642
~Gnss_Crypto
Gnss_Crypto, 608
~Gnss_Satellite
Gnss_Satellite, 619
~Gnss_Synchro
Gnss_Synchro, 633
~Nav_Message_Packet
Nav_Message_Packet, 759
~Nmea_Printer
Nmea_Printer, 767
~Rinex_Printer
Rinex_Printer, 882
~Rtcm_Printer
Rtcm_Printer, 900
~Secure_String
Secure_String, 944
~SignalConditioner
SignalConditioner, 964
~SignalSourceInterface
GNSS block interfaces, 84
~beidou_dnav_telemetry_decoder_gs
beidou_dnav_telemetry_decoder_gs, 394
~channel_msg_receiver_cc
channel_msg_receiver_cc, 408
~channel_status_msg_receiver
channel_status_msg_receiver, 408
~dll_pll_veml_tracking_fpga
dll_pll_veml_tracking_fpga, 459
~galileo_e5a_noncoherentIQ_acquisition_caf_cc
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 531
~galileo_e6_has_msg_receiver
galileo_e6_has_msg_receiver, 533
~galileo_pcps_8ms_acquisition_cc
galileo_pcps_8ms_acquisition_cc, 562
~galileo_tow_map
galileo_tow_map, 569
~glonass_gnav_telemetry_decoder_gs
glonass_gnav_telemetry_decoder_gs, 598
~gnss_synchro_monitor
gnss_synchro_monitor, 638
~nav_message_monitor
nav_message_monitor, 758
~osnma_msg_receiver
osnma_msg_receiver, 799
~pcps_acquisition_fine_doppler_cc
pcps_acquisition_fine_doppler_cc, 812
~pcps_acquisition_fpga
pcps_acquisition_fpga, 814
~pcps_assisted_acquisition_cc
pcps_assisted_acquisition_cc, 817
~pcps_cccwsr_acquisition_cc
pcps_cccwsr_acquisition_cc, 819
~pcps_opencl_acquisition_cc
pcps_opencl_acquisition_cc, 822
~pcps_quicksync_acquisition_cc
pcps_quicksync_acquisition_cc, 824
~pcps_tong_acquisition_cc
pcps_tong_acquisition_cc, 827
~rtklib_pvt_gs
rtklib_pvt_gs, 912

A
alm_t, 352
eph_t, 471
A0
Beidou_Cnav1_Utc_Model, 376
Galileo_Utc_Model, 570
Gnss_Ephemeris, 613

Gps_Utc_Model, 680
A0_6_LSB
core_system_parameters, 135
a0_bgto_s
Bds3_B1c_Bgto, 361
A0_GAL
Beidou_Dnav_Utc_Model, 396
A0_GLO
Beidou_Dnav_Utc_Model, 396
A0_GPS
Beidou_Dnav_Utc_Model, 396
A0_UTC
Beidou_Dnav_Utc_Model, 396
A1
Beidou_Cnav1_Utc_Model, 376
Galileo_Utc_Model, 570
Gps_Utc_Model, 680
A1_6_LSB
core_system_parameters, 135
a1_bgto_s_s
Bds3_B1c_Bgto, 361
A1_GAL
Beidou_Dnav_Utc_Model, 396
A1_GLO
Beidou_Dnav_Utc_Model, 397
A1_GPS
Beidou_Dnav_Utc_Model, 397
A1_UTC
Beidou_Dnav_Utc_Model, 397
A2
Beidou_Cnav1_Utc_Model, 376
Gps_Utc_Model, 680
a2_bgto_s_s2
Bds3_B1c_Bgto, 361
A_0_LSB
core_system_parameters, 135
A_0G
Galileo_Utc_Model, 570
A_0G_10_LSB
core_system_parameters, 135
A_1_LSB
core_system_parameters, 135
A_1_LSB_GAL
core_system_parameters, 135
A_1G
Galileo_Utc_Model, 570
A_1G_10_LSB
core_system_parameters, 135
A_F0_LSB
core_system_parameters, 136
A_F1_LSB
core_system_parameters, 136
A_F2_LSB
core_system_parameters, 136
ABS
telemetry_decoder_libswiftcnav, 314
ABSL_DECLARE_FLAG
gnss_sdr_flags, 256–259
acc
geph_t, 578
lexeph_t, 733
seph_t, 951
Acq_Conf, 331
acquire_iq, 332
acquire_pilot, 332
bit_transition_flag, 332
blocking, 333
blocking_on_standby, 333
cboc, 333
chips_per_second, 333
code_length, 333
cuda_device, 333
doppler_max, 333
doppler_min, 333
doppler_step, 333
doppler_step2, 333
dump, 333
dump_channel, 333
dump_filename, 334
enable_monitor_output, 334
folding_factor, 334
fs_in, 334
full_grid_search, 334
GetDwellResidualSamples, 332
GetSamplesPerDwell, 332
it_size, 334
item_type, 334
make_2_steps, 334
max_dwells, 334
ms_per_code, 334
num_codes, 334
num_doppler_bins_step2, 334
pfa, 335
pfa2, 335
qmboc, 335
reference_bin_min_sidelobes, 335
resampled_fs, 335
resampler_latency_samples, 335
resampler_ratio, 335
sampled_ms, 335
samples_per_chip, 335
samples_per_code, 335
samples_per_ms, 335
threshold, 335
use_automatic_resampler, 336
use_CFAR_algorithm_flag, 336
use_cuda, 336
vector_length, 336
zero_padding, 336
acq_conf.h, 1074, 1075
Acq_Conf_Fpga, 336
all_fft_codes, 337
code_length, 337
code_length_chips, 337
code_rate_cps, 337
device_name, 337

doppler_max, 337
doppler_step, 337
doppler_step2, 337
downsampling_factor, 337
downsampling_filter_delay, 337
downsampling_filter_num, 338
enable_zero_padding, 338
excludelimit, 338
fft_size, 338
fs_in, 338
make_2_steps, 338
max_num_acqs, 338
num_doppler_bins_step2, 338
repeat_satellite, 338
resampled_fs, 338
threshold, 338
total_block_exp, 338
acq_conf_fpga.h, 1076, 1077
Acq_delay_samples
Gnss_Synchro, 634
Acq_doppler_hz
Gnss_Synchro, 634
Acq_doppler_step
Gnss_Synchro, 634
Acq_samplestamp_samples
Gnss_Synchro, 634
acquire_iq
Acq_Conf, 332
acquire_pilot
Acq_Conf, 332
Acquisition, 79
acquisition
Channel, 405
acquisition_adapters, 79
acquisition_gr_blocks, 80
galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr, 81
galileo_pcps_8ms_acquisition_cc_sptr, 81
pcps_acquisition_fine_doppler_cc_sptr, 81
pcps_acquisition_fpga_sptr, 81
pcps_acquisition_sptr, 81
pcps_assisted_acquisition_cc_sptr, 81
pcps_cccwsr_acquisition_cc_sptr, 82
pcps_opencl_acquisition_cc_sptr, 82
pcps_quicksync_acquisition_cc_sptr, 82
pcps_tong_acquisition_cc_sptr, 82
acquisition_impl_interface, 339
acquisition_impl_interface, 340
mag, 340
set_active, 340
set_channel, 340
set_channel_fsm, 340
set_gnss_synchro, 340
set_local_code, 340, 341
acquisition_impl_interface.h, 1038, 1039
acquisition_impl_interface_sptr
GNSS block interfaces, 84
acquisition_interface.h, 1635, 1636
acquisition_libs, 82
acquisition_manager
GNSSFlowgraph, 642
AcquisitionInterface, 341
mag, 342
reset, 342
set_channel, 342
set_channel_fsm, 342
set_doppler_center, 342
set_doppler_num_bins, 342
set_gnss_synchro, 342
set_local_code, 343
set_resampler_latency, 343
stop_acquisition, 343
active_prn
Sbas_Rtklib_Corrections, 935
ad9361_manager.h, 1486, 1487
ad936x_custom_signal_source.h, 1395, 1396
ad936x_iio_custom, 343
n_channels, 344
ad936x_iio_custom.h, 1489
ad936x_iio_samples, 344
buffer, 344
n_bytes, 344
n_channels, 344
n_interleaved_iq_samples, 344
step_bytes, 344
ad936x_iio_samples.h, 1491, 1492
IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES, 1492
IIO_INPUTRAMFIFOSIZE, 1492
ad936x_iio_source, 344
start, 345
stop, 345
ad936x_iio_source.h, 1457, 1458
ad936x_iio_source_sptr
signal_source_gr_blocks, 307
Ad936xCustomSignalSource, 345
connect, 346
disconnect, 346
get_left_block, 347
get_right_block, 347
item_size, 347
add
beidou_ldpc.h, 1776
add_ms
galileo_tow_utils.h, 1544
gnss_tow, 324
addr
ftp_t, 519
tcp_t, 994
ADKD
MACK_tag_info, 741
Tag, 991
Adot
eph_t, 471
Gnss_Ephemeris, 613
adrv9361_z7035_signal_source_fpga.h, 1398

Adrv9361z7035SignalSourceFPGA, 347
connect, 348
disconnect, 348
get_left_block, 348
get_right_block, 348
item_size, 348
af0
Gnss_Almanac, 605
Gnss_Ephemeris, 613
lexeph_t, 733
seph_t, 951
af0_10
Galileo_Almanac_Helper, 524
AF0_10_LSB
core_system_parameters, 136
AF0_4_LSB
core_system_parameters, 136
af0_8
Galileo_Almanac_Helper, 524
AF0_8_LSB
core_system_parameters, 136
af0_9
Galileo_Almanac_Helper, 524
AF0_9_LSB
core_system_parameters, 136
af0_s
Bds3_B1c_AlmanacMedium, 359
af0red
Galileo_Reduced_CED, 565
af1
Gnss_Almanac, 605
Gnss_Ephemeris, 614
lexeph_t, 733
seph_t, 951
af1_10
Galileo_Almanac_Helper, 524
AF1_10_LSB
core_system_parameters, 136
AF1_4_LSB
core_system_parameters, 136
af1_8
Galileo_Almanac_Helper, 524
AF1_8_LSB
core_system_parameters, 136
af1_9
Galileo_Almanac_Helper, 524
AF1_9_LSB
core_system_parameters, 136
af1_s_s
Bds3_B1c_AlmanacMedium, 359
af1red
Galileo_Reduced_CED, 565
af2
Gnss_Ephemeris, 614
AF2_4_LSB
core_system_parameters, 136
age
geph_t, 578
sol_t, 971
age_s
Ntrip_Rtcm_Snapshot, 782
Agnss_Ref_Location, 349
Agnss_Ref_Location, 349
lat, 349
lon, 349
serialize, 349
uncertainty, 349
valid, 349
agnss_ref_location.h, 1720, 1721
Agnss_Ref_Time, 350
Agnss_Ref_Time, 350
microseconds, 350
seconds, 350
serialize, 350
tow, 351
valid, 351
week, 351
agnss_ref_time.h, 1722, 1723
ai
sbsfcorr_t, 936
ai0
Galileo_Iono, 559
AI0_5_LSB
core_system_parameters, 137
ai1
Galileo_Iono, 559
AI1_5_LSB
core_system_parameters, 137
ai2
Galileo_Iono, 559
AI2_5_LSB
core_system_parameters, 137
aif
Bds3_B1c_PageCommon, 363
alert
cnav_msg_t, 416
lexmsg_t, 735
alert_flag
Beidou_Dnav_Ephemeris, 386
Gps_CNAV_Ephemeris, 655
Gps_Ephemeris, 666
Algorithms Common Library, 231
algorithms_libs, 231
at, 238
back, 238
beidou_b1i_code_gen_complex, 239
beidou_b1i_code_gen_complex_sampled, 239
beidou_b1i_code_gen_float, 239
beidou_b1i_code_gen_int, 239
beidou_b3i_code_gen_complex, 239
beidou_b3i_code_gen_complex_sampled, 239
beidou_b3i_code_gen_float, 240
beidou_b3i_code_gen_int, 240
byte_x2_to_complex_byte_sptr, 238
cart2geo, 240
cart2utm, 240

clear, 240
clksin, 241
clsin, 241
complex_byte_to_float_x2_sptr, 238
complex_exp_gen, 241
complex_exp_gen_conj, 241
complex_float_to_complex_byte_sptr, 238
conjugate_cc_sptr, 238
conjugate_ic_sptr, 238
conjugate_sc_sptr, 238
cshort_to_float_x2_sptr, 238
findUtmZone, 241
front, 241
galileo_e1_code_gen_complex_sampled, 242
galileo_e1_code_gen_float_sampled, 242
galileo_e1_code_gen_sinboc11_float, 243
galileo_e5_a_code_gen_complex_primary, 243
galileo_e5_a_code_gen_complex_sampled, 243
galileo_e5_b_code_gen_complex_primary, 243
galileo_e5_b_code_gen_complex_sampled, 243
galileo_e6_b_code_gen_complex_primary, 243
galileo_e6_b_code_gen_complex_sampled, 244
galileo_e6_b_code_gen_float_primary, 244
galileo_e6_c_code_gen_complex_primary, 244
galileo_e6_c_code_gen_complex_sampled, 244
galileo_e6_c_code_gen_float_primary, 244
galileo_e6_c_secondary_code, 244
galileo_e6_c_secondary_code_gen_complex, 245
galileo_e6_c_secondary_code_gen_float, 245
Geo_to_ECEF, 245
get, 245
glonass_l1_ca_code_gen_complex, 245
glonass_l1_ca_code_gen_complex_sampled, 246
glonass_l1_ca_code_gen_float, 246
glonass_l2_ca_code_gen_complex, 246
glonass_l2_ca_code_gen_complex_sampled, 246
glonass_l2_ca_code_gen_float, 246
Gnss_circular_deque, 246
gps_l1_ca_code_gen_complex, 247
gps_l1_ca_code_gen_complex_sampled, 247
gps_l1_ca_code_gen_float, 247
gps_l1_ca_code_gen_int, 247
gps_l2c_m_code_gen_complex, 247
gps_l2c_m_code_gen_complex_sampled, 247, 248
gps_l2c_m_code_gen_float, 248
gps_l2c_m_code_gen_float_cl_zeroed, 248
gps_l5i_code_gen_complex, 248
gps_l5i_code_gen_complex_sampled, 248
gps_l5i_code_gen_float, 248
gps_l5q_code_gen_complex, 249
gps_l5q_code_gen_complex_sampled, 249
gps_l5q_code_gen_float, 249
Gravity_ECEF, 249
great_circle_distance, 249
hex_to_binary_converter, 249
hex_to_binary_string, 250
item_type_converter_t, 238
item_type_is_complex, 250
item_type_size, 250
item_type_valid, 250
make_vector_converter, 250
pop_front, 251
push_back, 251
pv_Geo_to_ECEF, 251
qzss_l1_code_gen_complex_sampled, 251
qzss_l1_code_gen_float, 252
qzss_l1_code_is_boc, 252
qzss_l5i_code_gen_complex_sampled, 252
qzss_l5i_code_gen_float, 252
qzss_l5q_code_gen_complex_sampled, 252
qzss_l5q_code_gen_float, 252
resample_sensor_data_tags, 252
resampler, 253
reset, 253
sbas_l1_code_gen_complex, 253
sbas_l1_code_gen_complex_sampled, 254
sbas_l1_code_gen_float, 254
short_x2_to_cshort_sptr, 238
size, 254
Skew_symmetric, 254
togeod, 254
topocent, 255
algorithms_libs_rtklib, 259
alm_to_rtklib, 277
ARMODE_CONT, 278
ARMODE_FIXHOLD, 278
ARMODE_INST, 278
ARMODE_OFF, 279
ARMODE_PPPAR, 279
ARMODE_PPPAR_ILS, 279
ARMODE_TCAR, 279
ARMODE_WLNL, 279
CHISQR, 279
closesocket, 271
COMMENTH, 271
dev_t, 271
DTTOL, 279
EFACT_BDS, 279
EFACT_GAL, 279
EFACT_GLO, 280
EFACT_GPS, 280
EFACT_IRN, 280
EFACT_QZS, 280
EFACT_SBS, 280
ENABDS, 271
ENAGLO, 271
ENAQZS, 271
eph_to_rtklib, 277
EPHOPT_BRDC, 280
EPHOPT_LEX, 280
EPHOPT_PREC, 280
EPHOPT_SBAS, 280
EPHOPT_SSRAPC, 281
EPHOPT_SSRCOM, 281
ERR_BRDCI, 281

ERR_CBIAS, 281
ERR_SAAS, 281
fatalfunc_t, 277
FE_WGS84, 281
FILEPATHSEP, 281
FTP_CMD, 271
FTP_TIMEOUT, 281
GAP_RESION, 281
glonass_almanac_epoch, 277
glonass_gnav_is_healthy, 278
HION, 281
IB_PPP, 271
IB_RTK, 272
IC_PPP, 272
II_RTK, 272
IL_RTK, 272
INIT_ZWD, 282
initlock, 272
INT_SWAP_STAT, 282
INT_SWAP_TRAC, 282
IONOOPT_BRDC, 282
IONOOPT_EST, 282
IONOOPT_IFLC, 282
IONOOPT_LEX, 282
IONOOPT_OFF, 282
IONOOPT_QZS, 282
IONOOPT_SBAS, 282
IONOOPT_STEC, 283
IONOOPT_TEC, 283
IT_PPP, 272
IT_RTK, 273
LAM_CARR, 283
lock_t, 273
MAXAC, 283
MAXANT, 283
MAXBAND, 283
MAXCLI, 283
MAXDTOE, 283
MAXDTOE_BDS, 283
MAXDTOE_GAL, 283
MAXDTOE_GLO, 284
MAXDTOE_QZS, 284
MAXDTOE_S, 284
MAXDTOE_SBS, 284
MAXERRMSG, 284
MAXEXFILE, 284
MAXFREQ, 284
MAXGDOP, 284
MAXLEAPS, 284
MAXNGEO, 284
MAXNIGP, 285
MAXOBS, 285
MAXOBSBUF, 285
MAXOBSTYPE, 285
MAXPRNBDS, 285
MAXPRNGAL, 285
MAXPRNGLO, 285
MAXPRNGPS, 285
MAXPRNIRN, 285
MAXPRNLEO, 285
MAXPRNQZS, 286
MAXPRNQZS_S, 286
MAXPRNSBS, 286
MAXRAWLEN, 286
MAXRCV, 286
MAXSAT, 286
MAXSBSAGEF, 286
MAXSBSAGEL, 286
MAXSBSMSG, 286
MAXSBSURA, 286
MAXSOLBUF, 287
MAXSOLMSG, 287
MAXSOLQ, 287
MAXSTA, 287
MAXSTATMSG, 287
MAXSTRMSG, 287
MAXSTRPATH, 287
MAXSTRRTK, 287
MINPRNBDS, 287
MINPRNGAL, 287
MINPRNGLO, 288
MINPRNGPS, 288
MINPRNIRN, 288
MINPRNLEO, 288
MINPRNQZS, 288
MINPRNQZS_S, 288
MINPRNSBS, 288
MSG_DISCONN, 273
NB_RTK, 273
NEXOBS, 288
NF_RTK, 273
NFREQ, 288
NFREQGLO, 288
NI_RTK, 273
NL_RTK, 273
NP_PPP, 273
NP_RTK, 273
NR_PPP, 274
NR_RTK, 274
NSATBDS, 289
NSATGAL, 289
NSATGLO, 289
NSATGPS, 289
NSATIRN, 289
NSATLEO, 289
NSATQZS, 289
NSATSBS, 289
NSYS, 289
NSYSBDS, 289
NSYSGAL, 290
NSYSGLO, 290
NSYSGPS, 290
NSYSIRN, 290
NSYSLEO, 290
NSYSQZS, 290
NT_RTK, 274

NTRIP_AGENT, 274
NTRIP_CLI_PORT, 274
NTRIP_MAXRSP, 274
NTRIP_MAXSTR, 274
NTRIP_RSP_ERROR, 274
NTRIP_RSP_HTTP, 274
NTRIP_RSP_OK_CLI, 274
NTRIP_RSP_OK_SVR, 275
NTRIP_RSP_SRCTBL, 275
NTRIP_RSP_TBLEND, 275
NTRIP_SVR_PORT, 275
NX_PPP, 275
NX_RTK, 275
PMODE_DGPS, 290
PMODE_FIXED, 290
PMODE_KINEMA, 290
PMODE_MOVEB, 290
PMODE_PPP_FIXED, 290
PMODE_PPP_KINEMA, 291
PMODE_PPP_STATIC, 291
PMODE_SINGLE, 291
PMODE_STATIC, 291
POLYCRC24Q, 291
POLYCRC32, 291
POSOPT_RINEX, 291
PRN_HWBIA, 291
PRN_HWBIAS, 291
RE_WGS84, 291
REL_HUMI, 292
rtk_lock, 275
rtk_unlock, 275
SERIBUFFSIZE, 292
socket_t, 275
SOLF_ENU, 292
SOLF_GSIF, 292
SOLF_LLH, 292
SOLF_NMEA, 292
SOLF_STAT, 292
SOLF_XYZ, 292
SOLQ_DGPS, 292
SOLQ_DR, 292
SOLQ_FIX, 293
SOLQ_FLOAT, 293
SOLQ_NONE, 293
SOLQ_PPP, 293
SOLQ_SBAS, 293
SOLQ_SINGLE, 293
STR_FILE, 275
STR_FTP, 276
STR_HTTP, 276
STR_MODE_R, 276
STR_MODE_RW, 276
STR_MODE_W, 276
STR_NONE, 276
STR_NTRIPCLI, 276
STR_NTRIPSVR, 276
STR_SERIAL, 276
STR_TCPCLI, 276
STR_TCPSVR, 276
STR_UDP, 276
STRFMT_NMEA, 293
STRFMT_RNXCLK, 293
STRFMT_RTCM2, 293
STRFMT_RTCM3, 293
STRFMT_SBAS, 293
STRFMT_SP3, 294
SYS_ALL, 294
SYS_BDS, 294
SYS_GAL, 294
SYS_GLO, 294
SYS_GPS, 294
SYS_IRN, 294
SYS_LEO, 294
SYS_NONE, 294
SYS_QZS, 295
SYS_SBS, 295
sysefact, 278
TIMES_GPST, 295
TIMES_JST, 295
TIMES_UTC, 295
TIMETAGH_LEN, 295
TINTACT, 295
TROPOPT_COR, 295
TROPOPT_CORG, 295
TROPOPT_EST, 295
TROPOPT_ESTG, 296
TROPOPT_OFF, 296
TROPOPT_SAAS, 296
TROPOPT_SBAS, 296
TTOL_MOVEB, 296
VAR_ACC, 296
VAR_GRA, 296
VAR_HOLDAMB, 296
VAR_HWBIAS, 296
VAR_POS, 296
VAR_VEL, 296
VER_RTKLIB, 277
alias
tled_t, 1002
all_fft_codes
Acq_Conf_Fpga, 337
Allocator, 351
Allocator, 351
buffer_, 352
countwords_, 352
operator(), 351
allowed_adkds
MACK_message, 740
Tag, 991
alm
nav_t, 762
ALM_AF0_LSB
core_system_parameters, 137
ALM_AF1_LSB
core_system_parameters, 137
ALM_DELTAI_LSB

core_system_parameters, 137
ALM_ECC_LSB
core_system_parameters, 137
ALM_MZERO_LSB
core_system_parameters, 137
ALM_OMEGA_LSB
core_system_parameters, 137
ALM_OMEGADOT_LSB
core_system_parameters, 137
ALM_OMEGAZERO_LSB
core_system_parameters, 137
ALM_SQUAREA_LSB
core_system_parameters, 137
alm_t, 352
A, 352
e, 352
f0, 352
f1, 352
i0, 352
M0, 353
omg, 353
OMG0, 353
OMGd, 353
sat, 353
svconf, 353
svh, 353
toa, 353
toas, 353
week, 353
alm_to_rtklib
algorithms_libs_rtklib, 277
ALM_TOA_LSB
core_system_parameters, 138
alpha
DSM_KROOT_message, 467
alpha0
Gps_Iono, 669
alpha1
Beidou_Cnav1_Iono, 373
Gps_Iono, 669
alpha2
Beidou_Cnav1_Iono, 373
Gps_Iono, 669
alpha3
Beidou_Cnav1_Iono, 374
Gps_Iono, 670
alpha4
Beidou_Cnav1_Iono, 374
alpha5
Beidou_Cnav1_Iono, 374
alpha6
Beidou_Cnav1_Iono, 374
alpha7
Beidou_Cnav1_Iono, 374
alpha8
Beidou_Cnav1_Iono, 374
alpha9
Beidou_Cnav1_Iono, 374
ALPHA_0
core_system_parameters, 133
ALPHA_0_LSB
core_system_parameters, 138
ALPHA_1_LSB
core_system_parameters, 138
ALPHA_2_LSB
core_system_parameters, 138
ALPHA_3_LSB
core_system_parameters, 138
always_output_gs
Obs_Conf, 787
ambc
rtk_t, 905
ambc_t, 354
epoch, 354
fixcnt, 354
flags, 354
LC, 354
LCv, 354
n, 354
AmEpID
Beidou_Dnav_Almanac, 380
AmID
Beidou_Dnav_Almanac, 380
an_dump_devname
Pvt_Conf, 856
an_output_enabled
Pvt_Conf, 856
An_Packet_Printer, 354
~An_Packet_Printer, 355
An_Packet_Printer, 355
close_serial, 355
print_packet, 355
an_packet_printer.h, 1303
an_packet_t, 355
data, 356
header, 356
an_rate_ms
Pvt_Conf, 856
announced_leap_second
Glonass_Gnav_Utc_Model, 600
antdel
prcopt_t, 845
antdes
sta_t, 985
antenna_height_m
Ntrip_Rtcm_Snapshot, 782
antispoofing_flag
Beidou_Dnav_Ephemeris, 386
Gps_CNAV_Ephemeris, 655
Gps_Ephemeris, 666
antsetup
sta_t, 985
antsno
sta_t, 985
anttype
prcopt_t, 845

AODC
Beidou_Dnav_Ephemeris, 386
AODE
Beidou_Dnav_Ephemeris, 386
AODO
Beidou_Dnav_Ephemeris, 386
Gps_Ephemeris, 666
AODO_LSB
core_system_parameters, 138
applicability
Osnma_Merkle_Tree_Material, 797
Osnma_Public_Key_Material, 804
applied_actions
ControlThread, 428
apply_action
GNSSFlowgraph, 642
apply_has_corrections
eph_t, 471
AR_ratio_factor
Monitor_Pvt, 750
AR_ratio_threshold
Monitor_Pvt, 750
arfilter
prcopt_t, 846
armaxiter
prcopt_t, 846
armaxposvar
prcopt_t, 846
ARMODE_CONT
algorithms_libs_rtklib, 278
ARMODE_FIXHOLD
algorithms_libs_rtklib, 278
ARMODE_INST
algorithms_libs_rtklib, 278
ARMODE_OFF
algorithms_libs_rtklib, 279
ARMODE_PPPAR
algorithms_libs_rtklib, 279
ARMODE_PPPAR_ILS
algorithms_libs_rtklib, 279
ARMODE_TCAR
algorithms_libs_rtklib, 279
ARMODE_WLNL
algorithms_libs_rtklib, 279
array_signal_conditioner.h, 1088, 1089
ArraySignalConditioner, 356
~ArraySignalConditioner, 357
ArraySignalConditioner, 357
connect, 357
data_type_adapter, 357
disconnect, 357
get_left_block, 357
get_right_block, 357
implementation, 357
input_filter, 357
item_size, 358
resampler, 358
role, 358
AS2R
core_system_parameters, 138
AS_status
Gps_Almanac, 650
assist_acquisition_doppler
Channel, 405
at
algorithms_libs, 238
AU
core_system_parameters, 138
available
Beidou_Dnav_Iono_Grid_Point, 390
az
solstat_t, 976
azel
ssat_t, 981
stec_t, 987
Azimuth
Gps_Acq_Assist, 646
azimuth_deg
Monitor_Pvt::TrackedSatelliteInfo, 1008

b1c_data_prompt_scale
Dll_Pll_Conf, 445
b1c_pilot_prompt_scale
Dll_Pll_Conf, 445
b1c_prompt_normalize_power
Dll_Pll_Conf, 445
b1c_prompt_use_data_q
Dll_Pll_Conf, 445
b1c_secondary_lock_ratio
Dll_Pll_Conf, 445
b_io_context
core_libs, 86
core_monitor, 89
pvt_libs, 299
signal_source_gr_blocks, 307
tracking_libs, 319
b_sv_do_not_use
Sbas_Ephemeris, 930
back
algorithms_libs, 238
bancroft_init
prcopt_t, 846
band_slot
Ntrip_Rtk_Signal, 784
base_position_ecef_m
Ntrip_Rtcm_Snapshot, 782
base_position_message_type
Ntrip_Rtcm_Snapshot, 782
baseline
prcopt_t, 846
bayesian_estimation.h, 1597, 1598
Bayesian_estimator, 358
BDEpID
Beidou_Dnav_Differential_Corrections, 382
Bds3_B1c_AlmanacMedium, 359
af0_s, 359
af1_s_s, 359

delta_i_rad, 359
eccentricity, 359
health, 359
m0_rad, 359
omega0_rad, 359
omega_dot_rad_s, 359
omega_rad, 360
prn, 360
sat_type, 360
sqrt_a_m_sqrt, 360
toa_s, 360
wna, 360
Bds3_B1c_AlmanacReduced, 360
delta_a_m, 360
health, 360
omega0_rad, 361
phi0_rad, 361
prn, 361
sat_type, 361
Bds3_B1c_Bgto, 361
a0_bgto_s, 361
a1_bgto_s_s, 361
a2_bgto_s_s2, 361
gnss_id, 361
t0_bgto_s, 362
wn0_bgto, 362
Bds3_B1c_Eop, 362
delta_ut1_dot_s_day, 362
delta_ut1_s, 362
pm_x_arcsec, 362
pm_x_dot_arcsec_day, 362
pm_y_arcsec, 362
pm_y_dot_arcsec_day, 362
t_eop_s, 363
Bds3_B1c_PageCommon, 363
aif, 363
dif, 363
hs, 363
page_id, 363
sif, 363
sismai, 363
Bds3_B1c_PageData, 364
bgto, 364
common, 364
eop, 364
medium_almanac, 364
reduced_almanac, 364
sisaioc, 364
sisaioe, 364
toa_s, 365
wna, 365
BDS_EPH_SELECTION_CNAV1_PREFERRED
rtklib_ephemeris.h, 1213
BDS_EPH_SOURCE_CNAV1
core_system_parameters, 138
BDS_EPH_SOURCE_CNAV2
core_system_parameters, 138
bdsmodear
prcopt_t, 846
beamformer, 365
beamformer.h, 1117, 1118
beamformer_filter.h, 1107
beamformer_sptr
input_filter_gr_blocks, 231
BeamformerFilter, 365
connect, 366
disconnect, 366
get_left_block, 366
get_right_block, 366
implementation, 366
item_size, 367
role, 367
Beidou_B1C.h, 1724, 1725
BEIDOU_B1C_CODE_LENGTH_CHIPS
core_system_parameters, 138
BEIDOU_B1C_CODE_PERIOD_MS
core_system_parameters, 139
BEIDOU_B1C_CODE_PERIOD_S
core_system_parameters, 139
BEIDOU_B1C_CODE_RATE_CPS
core_system_parameters, 139
Beidou_B1C_codes.h, 1726
BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH, 1726
BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH, 1726
BEIDOU_B1C_DATA_PRN_P
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_DATA_PRN_W
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_FRAME_LENGTH_SYMBOLS
core_system_parameters, 139
BEIDOU_B1C_FRAME_PERIOD_S
core_system_parameters, 139
BEIDOU_B1C_FREQ_HZ
core_system_parameters, 139
BEIDOU_B1C_NUMBER_OF_PRNS
core_system_parameters, 139
BEIDOU_B1C_OPT_ACQ_FS_SPS
core_system_parameters, 139
BEIDOU_B1C_PILOT_PRN_P
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_PILOT_PRN_W
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH
core_system_parameters, 139
BEIDOU_B1C_PILOT_SECONDARY_CODE_STR_LENGTH
Beidou_B1C_codes.h, 1726
BEIDOU_B1C_PILOT_SECONDARY_P
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_PILOT_SECONDARY_W
Beidou_B1C_prn.h, 1727
BEIDOU_B1C_PRIMARY_CODE_STR_LENGTH
Beidou_B1C_codes.h, 1726
Beidou_B1C_prn.h, 1727, 1728
BEIDOU_B1C_DATA_PRN_P, 1727

BEIDOU_B1C_DATA_PRN_W, 1727
BEIDOU_B1C_PILOT_PRN_P, 1727
BEIDOU_B1C_PILOT_PRN_W, 1727
BEIDOU_B1C_PILOT_SECONDARY_P, 1727
BEIDOU_B1C_PILOT_SECONDARY_W, 1727
beidou_b1c_signal_replica.h, 1123, 1124
BEIDOU_B1C_SYMBOLS_PER_BIT
core_system_parameters, 139
beidou_b1c_telemetry_decoder_gs, 367
reset, 368
set_channel, 368
set_satellite, 368
beidou_b1c_telemetry_decoder_gs.h, 1521, 1522
beidou_b1c_telemetry_decoder_gs_sptr, 1522
beidou_b1c_telemetry_decoder_gs_sptr
beidou_b1c_telemetry_decoder_gs.h, 1522
BEIDOU_B1C_TELEMETRY_RATE_SPS
core_system_parameters, 140
Beidou_B1I.h, 1732, 1733
beidou_b1i_code_gen_complex
algorithms_libs, 239
beidou_b1i_code_gen_complex_sampled
algorithms_libs, 239
beidou_b1i_code_gen_float
algorithms_libs, 239
beidou_b1i_code_gen_int
algorithms_libs, 239
BEIDOU_B1I_CODE_LENGTH_CHIPS
core_system_parameters, 140
BEIDOU_B1I_CODE_PERIOD_MS
core_system_parameters, 140
BEIDOU_B1I_CODE_PERIOD_S
core_system_parameters, 140
BEIDOU_B1I_CODE_RATE_CPS
core_system_parameters, 140
BEIDOU_B1I_D2_SECONDARY_CODE_STR
core_system_parameters, 140
BEIDOU_B1I_FREQ_HZ
core_system_parameters, 140
BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 140
BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR
core_system_parameters, 140
BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 140
BEIDOU_B1I_OPT_ACQ_FS_SPS
core_system_parameters, 140
BEIDOU_B1I_PREAMBLE_DURATION_MS
core_system_parameters, 141
BEIDOU_B1I_PREAMBLE_DURATION_S
core_system_parameters, 141
BEIDOU_B1I_PREAMBLE_LENGTH_BITS
core_system_parameters, 141
BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 141
BEIDOU_B1I_SECONDARY_CODE_LENGTH
core_system_parameters, 141
BEIDOU_B1I_SECONDARY_CODE_STR
core_system_parameters, 141
beidou_b1i_signal_replica.h, 1125
BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 141
BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND
core_system_parameters, 141
BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS
core_system_parameters, 141
BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 141
Beidou_B2a.h, 1734, 1735
BEIDOU_B2A_CODE_LENGTH_CHIPS
core_system_parameters, 141
BEIDOU_B2A_CODE_PERIOD_MS
core_system_parameters, 142
BEIDOU_B2A_CODE_PERIOD_S
core_system_parameters, 142
BEIDOU_B2A_CODE_RATE_CPS
core_system_parameters, 142
BEIDOU_B2A_FREQ_HZ
core_system_parameters, 142
BEIDOU_B2A_OPT_ACQ_FS_SPS
core_system_parameters, 142
BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 142
BEIDOU_B2A_PREAMBLE_SYMBOLS_STR
core_system_parameters, 142
BEIDOU_B2A_SECONDARY_CODE_LENGTH
core_system_parameters, 142
BEIDOU_B2A_SECONDARY_CODE_STR
core_system_parameters, 142
beidou_b2a_signal_replica.h, 1126
BEIDOU_B2A_SYMBOLS_PER_BIT
core_system_parameters, 142
beidou_b2a_telemetry_decoder_gs, 368
reset, 369
set_channel, 369
set_satellite, 369
beidou_b2a_telemetry_decoder_gs.h, 1523, 1524
beidou_b2a_telemetry_decoder_gs_sptr, 1523
beidou_b2a_telemetry_decoder_gs_sptr
beidou_b2a_telemetry_decoder_gs.h, 1523
BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND
core_system_parameters, 142
Beidou_B3I.h, 1735, 1736
beidou_b3i_code_gen_complex
algorithms_libs, 239
beidou_b3i_code_gen_complex_sampled
algorithms_libs, 239
beidou_b3i_code_gen_float
algorithms_libs, 240
beidou_b3i_code_gen_int
algorithms_libs, 240
BEIDOU_B3I_CODE_LENGTH_CHIPS
core_system_parameters, 143
BEIDOU_B3I_CODE_PERIOD_MS
core_system_parameters, 143
BEIDOU_B3I_CODE_PERIOD_S

core_system_parameters, 143
BEIDOU_B3I_CODE_RATE_CPS
core_system_parameters, 143
BEIDOU_B3I_D2_SECONDARY_CODE_STR
core_system_parameters, 143
BEIDOU_B3I_FREQ_HZ
core_system_parameters, 143
BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 143
BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR
core_system_parameters, 143
BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 143
BEIDOU_B3I_OPT_ACQ_FS_SPS
core_system_parameters, 143
BEIDOU_B3I_PREAMBLE_DURATION_MS
core_system_parameters, 143
BEIDOU_B3I_PREAMBLE_DURATION_S
core_system_parameters, 144
BEIDOU_B3I_PREAMBLE_LENGTH_BITS
core_system_parameters, 144
BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 144
BEIDOU_B3I_SECONDARY_CODE_LENGTH
core_system_parameters, 144
BEIDOU_B3I_SECONDARY_CODE_STR
core_system_parameters, 144
beidou_b3i_signal_replica.h, 1127, 1128
BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 144
BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND
core_system_parameters, 144
BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS
core_system_parameters, 144
BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 144
beidou_bdgim.h, 1737
beidou_bdgim_delay_m
core_system_parameters, 133
beidou_bdgim_nearest_odd_hour_mjd
core_system_parameters, 134
Beidou_CNAV1.h, 1738, 1739
BEIDOU_CNAV1_A_DOT_LSB
core_system_parameters, 144
BEIDOU_CNAV1_A_REF_IGSO
core_system_parameters, 144
BEIDOU_CNAV1_A_REF_MEO
core_system_parameters, 145
BEIDOU_CNAV1_AF0_LSB
core_system_parameters, 145
BEIDOU_CNAV1_AF1_LSB
core_system_parameters, 145
BEIDOU_CNAV1_AF2_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CIC_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CIS_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CRC_BITS
core_system_parameters, 145
BEIDOU_CNAV1_CRC_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CRS_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CUC_LSB
core_system_parameters, 145
BEIDOU_CNAV1_CUS_LSB
core_system_parameters, 145
BEIDOU_CNAV1_DELTA_A_LSB
core_system_parameters, 145
BEIDOU_CNAV1_DELTA_N0_DOT_LSB
core_system_parameters, 146
BEIDOU_CNAV1_DELTA_N0_LSB
core_system_parameters, 146
BEIDOU_CNAV1_E_LSB
core_system_parameters, 146
Beidou_Cnav1_Ephemeris, 369
Beidou_Cnav1_Ephemeris, 372
hs, 372
IODC, 372
IODE, 372
ISC_B1Cd, 372
ISC_B2ad, 372
nav_type, 372
sat_type, 373
serialize, 372
sig_type, 373
TGD_B1Cp, 373
TGD_B2ap, 373
beidou_cnav1_ephemeris.h, 1740, 1741
beidou_cnav1_ephemeris_map
Rtklib_Solver, 918
BEIDOU_CNAV1_FRAME_PERIOD_S
core_system_parameters, 146
BEIDOU_CNAV1_FRAME_SYMBOLS
core_system_parameters, 146
BEIDOU_CNAV1_H100_200_ELEMENT
core_system_parameters, 146
BEIDOU_CNAV1_H100_200_INDEX
core_system_parameters, 146
BEIDOU_CNAV1_H44_88_ELEMENT
core_system_parameters, 147
BEIDOU_CNAV1_H44_88_INDEX
core_system_parameters, 147
BEIDOU_CNAV1_I0_LSB
core_system_parameters, 147
BEIDOU_CNAV1_IDOT_LSB
core_system_parameters, 147
BEIDOU_CNAV1_INTERLEAVE_COLS
core_system_parameters, 147
BEIDOU_CNAV1_INTERLEAVE_ROWS
core_system_parameters, 147
BEIDOU_CNAV1_INTERLEAVED_SYMBOLS
core_system_parameters, 147
Beidou_Cnav1_Iono, 373
alpha1, 373

alpha2, 373
alpha3, 374
alpha4, 374
alpha5, 374
alpha6, 374
alpha7, 374
alpha8, 374
alpha9, 374
valid, 374
beidou_cnav1_iono
Rtklib_Solver, 918
beidou_cnav1_iono.h, 1742
BEIDOU_CNAV1_ISC_LSB
core_system_parameters, 148
beidou_cnav1_ldpc.h, 1743, 1744
BEIDOU_CNAV1_LDPC_DC200
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_DC88
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_K200
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_K88
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_M200
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_M88
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_MAX_ITER
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_N200
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_N88
core_system_parameters, 148
BEIDOU_CNAV1_LDPC_NM
core_system_parameters, 148
BEIDOU_CNAV1_M0_LSB
core_system_parameters, 148
Beidou_Cnav1_Navigation_Message, 374
get_last_nav_bits, 375
beidou_cnav1_navigation_message.h, 1745, 1746
BEIDOU_CNAV1_OMEGA_LSB
core_system_parameters, 149
BEIDOU_CNAV1_OMEGADOT_LSB
core_system_parameters, 149
beidou_cnav1_page_data_map
Rtklib_Solver, 918
BEIDOU_CNAV1_PAGE_IONO_UTC
core_system_parameters, 149
Beidou_Cnav1_PageData_Message, 375
page_data, 375
PRN, 375
BEIDOU_CNAV1_SF2_DATA_BITS
core_system_parameters, 149
BEIDOU_CNAV1_SF3_DATA_BITS
core_system_parameters, 149
BEIDOU_CNAV1_SOH_LSB_S
core_system_parameters, 149
BEIDOU_CNAV1_SUBFRAME1_SYMBOLS
core_system_parameters, 149
BEIDOU_CNAV1_SUBFRAME2_SYMBOLS
core_system_parameters, 149
BEIDOU_CNAV1_SUBFRAME3_SYMBOLS
core_system_parameters, 149
BEIDOU_CNAV1_SYMBOL_RATE_SPS
core_system_parameters, 149
BEIDOU_CNAV1_TGD_LSB
core_system_parameters, 149
BEIDOU_CNAV1_TOE_TOC_LSB
core_system_parameters, 149
Beidou_Cnav1_Utc_Model, 375
A0, 376
A1, 376
A2, 376
delta_t_LS, 376
delta_t_LSF, 376
DN, 376
tot, 376
valid, 376
WN_LSF, 376
WN_t, 377
beidou_cnav1_utc_model
Rtklib_Solver, 918
beidou_cnav1_utc_model.h, 1748
Beidou_CNAV2.h, 1749, 1750
BEIDOU_CNAV2_CRC_BITS
core_system_parameters, 150
BEIDOU_CNAV2_DATA_BITS
core_system_parameters, 150
beidou_cnav2_ephemeris_map
Rtklib_Solver, 918
BEIDOU_CNAV2_FRAME_MS
core_system_parameters, 150
BEIDOU_CNAV2_FRAME_PERIOD_S
core_system_parameters, 150
BEIDOU_CNAV2_FRAME_SYMBOLS
core_system_parameters, 150
BEIDOU_CNAV2_H48_96_ELEMENT
core_system_parameters, 150
BEIDOU_CNAV2_H48_96_INDEX
core_system_parameters, 150
BEIDOU_CNAV2_INFO_BITS
core_system_parameters, 150
BEIDOU_CNAV2_IODE_BITS
core_system_parameters, 151
beidou_cnav2_ldpc.h, 1750, 1751
BEIDOU_CNAV2_LDPC_DC
core_system_parameters, 151
BEIDOU_CNAV2_LDPC_K
core_system_parameters, 151
BEIDOU_CNAV2_LDPC_M
core_system_parameters, 151
BEIDOU_CNAV2_LDPC_N
core_system_parameters, 151
BEIDOU_CNAV2_LDPC_SYMBOLS
core_system_parameters, 151
BEIDOU_CNAV2_MES_TYPE_BITS

core_system_parameters, 151
BEIDOU_CNAV2_MS_PER_SYMBOL
core_system_parameters, 151
BEIDOU_CNAV2_MSG_ALM
core_system_parameters, 151
BEIDOU_CNAV2_MSG_CLK_ALM
core_system_parameters, 151
BEIDOU_CNAV2_MSG_CLK_DC
core_system_parameters, 151
BEIDOU_CNAV2_MSG_CLK_EOP
core_system_parameters, 151
BEIDOU_CNAV2_MSG_CLK_IONO
core_system_parameters, 152
BEIDOU_CNAV2_MSG_CLK_UTC
core_system_parameters, 152
BEIDOU_CNAV2_MSG_EPH1
core_system_parameters, 152
BEIDOU_CNAV2_MSG_EPH2
core_system_parameters, 152
BEIDOU_CNAV2_MT10_IODE_BIT
core_system_parameters, 152
Beidou_Cnav2_Navigation_Message, 377
clear_flags, 377
get_ephemeris, 377
get_last_nav_bits, 377
have_new_ephemeris, 377
last_crc_ok, 377
last_frame_prn, 378
last_mes_type, 378
last_sow, 378
preamble_detection_statistic, 378
beidou_cnav2_navigation_message.h, 1752
BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD
core_system_parameters, 152
BEIDOU_CNAV2_PREAMBLE_MS
core_system_parameters, 152
BEIDOU_CNAV2_PREAMBLE_SYMBOLS
core_system_parameters, 152
BEIDOU_CNAV2_SOW_LSB_S
core_system_parameters, 152
BEIDOU_CNAV2_SYMBOL_RATE_SPS
core_system_parameters, 152
Beidou_DNAV.h, 1753, 1758
Beidou_Dnav_Almanac, 378
AmEpID, 380
AmID, 380
Beidou_Dnav_Almanac, 380
expanded, 380
serialize, 380
SV_health, 380
beidou_dnav_almanac.h, 1762
beidou_dnav_almanac_map
Rtklib_Solver, 918
BEIDOU_DNAV_BCH_CODE_BITS
core_system_parameters, 152
BEIDOU_DNAV_BCH_DATA_BITS
core_system_parameters, 152
BEIDOU_DNAV_BCH_PARITY_BITS
core_system_parameters, 153
BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET
core_system_parameters, 153
BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET
core_system_parameters, 153
BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS
core_system_parameters, 153
BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS
core_system_parameters, 153
Beidou_Dnav_Differential_Correction, 380
delta_t, 381
delta_t_available, 381
PRN, 381
RURAI, 381
UDREI, 381
Beidou_Dnav_Differential_Corrections, 381
BDEpID, 382
corrections, 382
expanded_corrections, 382
ionospheric_grid_valid, 382
Pnum2, 382
range_corrections_valid, 382
SatH2, 382
source_PRN, 382
SOW, 382
beidou_dnav_differential_corrections.h, 1763, 1764
Beidou_Dnav_Ephemeris, 383
alert_flag, 386
antispoofing_flag, 386
AODC, 386
AODE, 386
AODO, 386
Beidou_Dnav_Ephemeris, 385
fit_interval_flag, 386
integrity_status_flag, 387
nav_type, 387
satelliteBlock, 387
serialize, 386
sig_type, 387
spare1, 387
spare2, 387
SV_accuracy, 387
SV_health, 387
TGD1, 387
TGD2, 388
beidou_dnav_ephemeris.h, 1765
beidou_dnav_ephemeris_map
Rtklib_Solver, 918
Beidou_Dnav_Iono, 388
Beidou_Dnav_Iono, 389
grid_points, 389
grid_valid, 389
serialize, 389
beidou_dnav_iono
Rtklib_Solver, 918
beidou_dnav_iono.h, 1767
Beidou_Dnav_Iono_Grid_Point, 390
available, 390

GIVEI, 390
monitored, 390
serialize, 390
vertical_delay, 390
Beidou_Dnav_Navigation_Message, 391
Beidou_Dnav_Navigation_Message, 391
d1_subframe_decoder, 392
d2_subframe_decoder, 392
get_almanac, 392
get_differential_corrections, 392
get_ephemeris, 392
get_flag_CRC_test, 392
get_flag_new_SOW_available, 392
get_iono, 392
get_SOW, 392
get_utc_model, 392
have_new_almanac, 392
have_new_differential_corrections, 392
have_new_ephemeris, 393
have_new_iono, 393
have_new_utc_model, 393
set_flag_new_SOW_available, 393
set_satellite_PRN, 393
set_signal_type, 393
utc_time, 393
beidou_dnav_navigation_message.h, 1768, 1769
BEIDOU_DNAV_PREAMBLE
core_system_parameters, 153
BEIDOU_DNAV_PREAMBLE_LENGTH_BITS
core_system_parameters, 153
BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 153
BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS
core_system_parameters, 153
BEIDOU_DNAV_SECONDS_PER_WEEK
core_system_parameters, 153
BEIDOU_DNAV_SUBFRAME_DATA_BITS
core_system_parameters, 153
BEIDOU_DNAV_SUBFRAME_SYMBOLS
core_system_parameters, 153
beidou_dnav_telemetry_decoder_gs, 393
~beidou_dnav_telemetry_decoder_gs, 394
general_work, 395
reset, 395
set_channel, 395
set_satellite, 395
beidou_dnav_telemetry_decoder_gs.h, 1525
beidou_dnav_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
Beidou_Dnav_Utc_Model, 395
A0_GAL, 396
A0_GLO, 396
A0_GPS, 396
A0_UTC, 396
A1_GAL, 396
A1_GLO, 397
A1_GPS, 397
A1_UTC, 397
DeltaT_LS, 397
DeltaT_LSF, 397
DN, 397
serialize, 396
valid, 397
WN_LSF, 397
beidou_dnav_utc_model
Rtklib_Solver, 918
beidou_dnav_utc_model.h, 1774
BEIDOU_DNAV_WORD_LENGTH_BITS
core_system_parameters, 154
BEIDOU_DNAV_WORDS_SUBFRAME
core_system_parameters, 154
BEIDOU_F
core_system_parameters, 154
BEIDOU_GF64_EXP
beidou_ldpc.h, 1777
BEIDOU_GF64_LOG
beidou_ldpc.h, 1777
BEIDOU_GM
core_system_parameters, 154
beidou_ldpc.h, 1775, 1777
add, 1776
BEIDOU_GF64_EXP, 1777
BEIDOU_GF64_LOG, 1777
BEIDOU_LDPC_MAX_ITER, 1777
BEIDOU_LDPC_NM, 1777
inv, 1776
kFieldSize, 1777
kOrder, 1777
kZero, 1777
mul, 1776
valid_symbol, 1776
BEIDOU_LDPC_MAX_ITER
beidou_ldpc.h, 1777
BEIDOU_LDPC_NM
beidou_ldpc.h, 1777
BEIDOU_OMEGA_EARTH_DOT
core_system_parameters, 154
BeidouCnav1LdpcGraph
core_system_parameters, 132
BeidouLdpcGraph, 398
check_offsets, 398
check_to_h, 398
check_to_var, 398
num_checks, 398
num_variables, 398
row_weight, 398
var_offsets, 398
var_to_check, 398
var_to_edge, 399
var_to_h, 399
var_to_h_inv, 399
beta0
Gps_Iono, 670
beta1
Gps_Iono, 670
beta2

Gps_Iono, 670
beta3
Gps_Iono, 670
BETA_0_LSB
core_system_parameters, 154
BETA_1_LSB
core_system_parameters, 154
BETA_2_LSB
core_system_parameters, 154
BETA_3_LSB
core_system_parameters, 154
BGD_E1_E5A_5_LSB
core_system_parameters, 154
BGD_E1_E5B_5_LSB
core_system_parameters, 154
BGD_E1E5a
Galileo_Ephemeris, 536
BGD_E1E5b
Galileo_Ephemeris, 537
Galileo_Reduced_CED, 565
bgto
Bds3_B1c_PageData, 364
bias
fcbd_t, 481
HAS_bias_corrections, 697
bin_to_binary_data
Rtcm, 888
bin_to_double
Rtcm, 888
bin_to_hex
Rtcm, 888
bin_to_uint
Rtcm, 888
binary_data_to_bin
Rtcm, 888
bins
HistogramBitSynchronizer, 702
bit_offset_
IONGSMSChunkUnpackingCtx< WT >, 717
bit_period_ms
HistogramBitSynchronizer::Config, 423
bit_selection
Fpga_dynamic_bit_selection, 512
bit_synchronization_time_limit_s
Dll_Pll_Conf, 445
Dll_Pll_Conf_Fpga, 452
Kf_Conf, 726
bit_synchronizer.h, 1599, 1600
bit_transition_flag
Acq_Conf, 332
bite
sbsigpband_t, 938
bits
sbsigpband_t, 938
bits.h, 1545, 1546
BITS_IN_OCTET
core_system_parameters, 155
bits_to_item_size
ion_gsms_chunk_data.h, 1505
bladerf_signal_source.h, 1400
bladerf_source, 399
bladerf_source.h, 1460
bladerf_source_sptr
signal_source_gr_blocks, 308
BladerfSignalSource, 400
connect, 401
disconnect, 401
get_left_block, 401
get_right_block, 401
item_size, 401
block
IONGSMSFileSource::SegmentDescriptor, 945
block_extends_to_eof
IONGSMSFileSource::SegmentDescriptor, 945
block_factory, 323
get_role_name, 323
GetBlock, 324
block_start_offset
IONGSMSFileSource::SegmentDescriptor, 945
blocking
Acq_Conf, 333
blocking_on_standby
Acq_Conf, 333
bs_dominance_ratio
Dll_Pll_Conf, 445
Dll_Pll_Conf_Fpga, 452
bs_min_events_for_lock
Dll_Pll_Conf, 445
Dll_Pll_Conf_Fpga, 452
bs_min_prompt_mag
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_runner_up_margin
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_stable_best_required
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_tentative_events_required
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_transition_confidence
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_transition_window_epochs
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bs_use_phase_dot_detector
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 452
bstar
tled_t, 1002
buff
ntrip_t, 785
raw_t, 875
rtcm_t, 902

rtksvr_t, 922
solbuf_t, 973
buffer
ad936x_iio_samples, 344
fpga_dma-proxy.h, 1495
buffer_
Allocator, 352
Deleter, 439
buffsize
rtksvr_t, 922
bw_hz
stream_cfg, 988
by_source
Galileo_Ephemeris_Store, 539
byte_to_short.h, 1091, 1092
byte_x2_to_complex_byte, 401
byte_x2_to_complex_byte.h, 1128, 1129
byte_x2_to_complex_byte_sptr
algorithms_libs, 238
bytes_received
Ntrip_Rtcm_Client_Status, 781
ByteToShort, 402
connect, 402
disconnect, 402
get_left_block, 402
get_right_block, 402
implementation, 403
item_size, 403
role, 403

c0
v27_poly_t, 1027
c1
v27_poly_t, 1027
C_IC_4_LSB
core_system_parameters, 155
C_IC_LSB
core_system_parameters, 155
C_IS_4_LSB
core_system_parameters, 155
C_IS_LSB
core_system_parameters, 155
C_RC_3_LSB
core_system_parameters, 155
C_RC_LSB
core_system_parameters, 155
C_RS_3_LSB
core_system_parameters, 155
C_RS_LSB
core_system_parameters, 155
C_UC_3_LSB
core_system_parameters, 155
C_UC_LSB
core_system_parameters, 155
C_US_3_LSB
core_system_parameters, 155
C_US_LSB
core_system_parameters, 156
ca_codes
Dll_Pll_Conf_Fpga, 453
CalculateSlope
tracking_libs, 319
CalculateSlopeAbs
tracking_libs, 319
carrier_aiding
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 453
carrier_disc_sd_rads
Kf_Conf, 726
Carrier_Doppler_hz
Gnss_Synchro, 634
carrier_freq_hz
TrackingCmd, 1012
carrier_freq_rate_hz_s
TrackingCmd, 1012
carrier_freq_rate_sd_hz_s
Kf_Conf, 726
carrier_freq_sd_hz
Kf_Conf, 726
carrier_lock_detector
tracking_libs, 319
carrier_lock_test_smoother_alpha
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
carrier_lock_test_smoother_samples
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
carrier_lock_th
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
Carrier_phase_rads
Gnss_Synchro, 634
carrier_phase_sd_rad
Kf_Conf, 726
Carrier_wipeoff_multicorrelator_resampler
Fpga_Multicorrelator_8sc, 514
cart2geo
algorithms_libs, 240
cart2utm
algorithms_libs, 240
cbias
nav_t, 762
ssr_t, 983
cboc
Acq_Conf, 333
CED_af0red_LSB
core_system_parameters, 156
CED_af1red_LSB
core_system_parameters, 156
CED_DeltaAred_LSB
core_system_parameters, 156
CED_Deltai0red_LSB
core_system_parameters, 156
CED_exred_LSB

core_system_parameters, 156
CED_eyred_LSB
core_system_parameters, 156
CED_lambda0red_LSB
core_system_parameters, 156
CED_Omega0red_LSB
core_system_parameters, 156
CED_VALIDITY_SECONDS
Galileo_Reduced_CED, 565
cell_mask
Galileo_HAS_data, 550
cell_mask_availability_flag
Galileo_HAS_data, 550
cellmask
msm_h_t, 754
cerr
exterr_t, 480
Channel, 227, 403
~Channel, 404
acquisition, 405
assist_acquisition_doppler, 405
Channel, 404
connect, 405
disconnect, 405
get_left_block, 405
get_left_block_acq, 405
get_left_block_trk, 405
get_right_block, 405
get_right_block_acq, 405
get_right_block_trk, 405
get_signal, 405
implementation, 406
item_size, 406
role, 406
set_signal, 406
start_acquisition, 406
stop_channel, 406
telemetry, 406
tracking, 406
channel
TOW_to_trk, 1007
channel.h, 1083, 1084
channel_adapters, 227
Channel_Event, 407
channel_id, 407
event_type, 407
channel_event.h, 1647, 1648
channel_event_sptr
core_libs, 86
channel_fsm.h, 1085, 1086
Channel_ID
Gnss_Synchro, 634
channel_id
Channel_Event, 407
channel_interface.h, 1637
channel_libs, 227
channel_msg_receiver_cc_sptr, 227
channel_msg_receiver_cc, 407
~channel_msg_receiver_cc, 408
channel_msg_receiver_cc.h, 1087, 1088
channel_msg_receiver_cc_sptr
channel_libs, 227
channel_status_msg_receiver, 408
~channel_status_msg_receiver, 408
clear_channel_status, 409
get_current_status_map, 409
get_current_status_pvt, 409
get_current_tracking_map, 409
channel_status_msg_receiver.h, 1649
channel_status_msg_receiver_sptr
core_libs, 87
ChannelFsm, 409
ChannelInterface, 410
get_left_block, 410
get_left_block_acq, 410
get_left_block_trk, 411
get_right_block, 411
get_right_block_acq, 411
get_right_block_trk, 411
set_signal, 411
start_acquisition, 411
stop_channel, 411
check_any_enabled
Signal_Enabled_Flags, 961
check_buffer_overflow_and_monitor_buffer_status
Fpga_buffer_monitor, 510
check_CRC
Rtcm, 888
check_offsets
BeidouLdpcGraph, 398
check_only_enabled
Signal_Enabled_Flags, 961
check_to_h
BeidouLdpcGraph, 398
check_to_var
BeidouLdpcGraph, 398
chips_per_second
Acq_Conf, 333
CHISQR
algorithms_libs_rtklib, 279
chunk_remaining
ntrip_t, 785
chunk_size
SensorDataFileConfiguration, 947
chunk_state
ntrip_t, 785
chunked
ntrip_t, 785
Cic
Gnss_Ephemeris, 614
cic
eph_t, 471
cid
DSM_nma_header, 468
cidkr
DSM_KROOT_message, 467

circular_error_ms
gnss_tow, 324
Cis
Gnss_Ephemeris, 614
cis
eph_t, 471
cl_fft_plan, 411
context, 412
dim, 412
format, 412
kernel_info, 412
kernel_string, 412
last_batch_size, 412
max_localmem_fft_size, 412
max_radix, 412
max_work_item_per_workgroup, 412
min_mem_coalesce_width, 412
n, 413
num_kernels, 413
num_local_mem_banks, 413
program, 413
temp_buffer_needed, 413
tempmemobj, 413
tempmemobj_imag, 413
tempmemobj_real, 413
twist_kernel, 413
CL_SILENCE_DEPRECATION
pcps_opencl_acquisition_cc.h, 1064
CLAMP_DIFF
telemetry_decoder_libswiftcnav, 314
class_type
matlab_type_traits< double >, 742
matlab_type_traits< float >, 742
matlab_type_traits< int32_t >, 743
matlab_type_traits< int64_t >, 743
matlab_type_traits< uint32_t >, 744
matlab_type_traits< uint64_t >, 744
matlab_type_traits< uint8_t >, 745
clear
algorithms_libs, 240
Concurrent_Queue< Data >, 421
Galileo_Ephemeris_Store, 539
Secure_String, 944
clear_channel_status
channel_status_msg_receiver, 409
clear_ephemeris
Rtklib_Pvt, 909
rtklib_pvt_gs, 912
clear_ephemeris_keep_almanac
Rtklib_Pvt, 909
rtklib_pvt_gs, 912
clear_flags
Beidou_Cnav2_Navigation_Message, 377
clear_public_key
Gnss_Crypto, 608
clear_tag_accumulation_metadata
OSNMA_NavData, 801
clFFT.h, 1172, 1173
clFFT_DataFormat, 1173
clFFT_Dimension, 1173
clFFT_Direction, 1173
clFFT_Plan, 1173
clFFT_Complex, 413
imag, 414
real, 414
clFFT_DataFormat
clFFT.h, 1173
clFFT_Dim3, 414
x, 414
y, 414
z, 414
clFFT_Dimension
clFFT.h, 1173
clFFT_Direction
clFFT.h, 1173
clFFT_Plan
clFFT.h, 1173
clFFT_SplitComplex, 414
imag, 415
real, 415
cli
tcpsvr_t, 996
clk
pclk_t, 807
clk_ext
msm_h_t, 754
clk_str
msm_h_t, 754
clksin
algorithms_libs, 241
clock_correction_m
HAS_clock_corrections, 698
clock_fullset_flag
mt1_header, 755
clock_subset_flag
mt1_header, 755
close_device
Fpga_Acquisition, 508
close_serial
An_Packet_Printer, 355
closesocket
algorithms_libs_rtklib, 271
clsin
algorithms_libs, 241
CN0_dB_hz
Gnss_Synchro, 634
cn0_m2m4_estimator
tracking_libs, 319
cn0_min
Dll_Pll_Conf, 446
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
cn0_samples
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726

cn0_smoother_alpha
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
cn0_smoother_samples
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 453
Kf_Conf, 726
cn0_svn_estimator
tracking_libs, 320
CNAV_A0_LSB
core_system_parameters, 156
CNAV_A1_LSB
core_system_parameters, 156
CNAV_A2_LSB
core_system_parameters, 156
CNAV_A_DOT_LSB
core_system_parameters, 157
CNAV_A_REF
core_system_parameters, 157
CNAV_AF0_LSB
core_system_parameters, 157
CNAV_AF1_LSB
core_system_parameters, 157
CNAV_AF2_LSB
core_system_parameters, 157
CNAV_ALPHA0_LSB
core_system_parameters, 157
CNAV_ALPHA1_LSB
core_system_parameters, 157
CNAV_ALPHA2_LSB
core_system_parameters, 157
CNAV_ALPHA3_LSB
core_system_parameters, 157
CNAV_BETA0_LSB
core_system_parameters, 157
CNAV_BETA1_LSB
core_system_parameters, 157
CNAV_BETA2_LSB
core_system_parameters, 157
CNAV_BETA3_LSB
core_system_parameters, 158
CNAV_CIC_LSB
core_system_parameters, 158
CNAV_CIS_LSB
core_system_parameters, 158
CNAV_CRC_LSB
core_system_parameters, 158
CNAV_CRS_LSB
core_system_parameters, 158
CNAV_CUC_LSB
core_system_parameters, 158
CNAV_CUS_LSB
core_system_parameters, 158
CNAV_DELTA_A_LSB
core_system_parameters, 158
CNAV_DELTA_N0_DOT_LSB
core_system_parameters, 158
CNAV_DELTA_N0_LSB
core_system_parameters, 158
CNAV_DELTA_OMEGA_DOT_LSB
core_system_parameters, 158
CNAV_DELTA_TLS_LSB
core_system_parameters, 158
CNAV_DELTA_TLSF_LSB
core_system_parameters, 159
CNAV_DELTA_UT1_GPS_DOT_LSB
core_system_parameters, 159
CNAV_DELTA_UT1_GPS_LSB
core_system_parameters, 159
CNAV_DN_LSB
core_system_parameters, 159
CNAV_E_ECCENTRICITY_LSB
core_system_parameters, 159
CNAV_I0_DOT_LSB
core_system_parameters, 159
CNAV_I0_LSB
core_system_parameters, 159
CNAV_ISCL1_LSB
core_system_parameters, 159
CNAV_ISCL2_LSB
core_system_parameters, 159
CNAV_ISCL5I_LSB
core_system_parameters, 159
CNAV_ISCL5Q_LSB
core_system_parameters, 159
CNAV_M0_LSB
core_system_parameters, 159
cnav_msg.h, 1547, 1548
cnav_msg_decoder_t, 415
part1, 415
part2, 415
cnav_msg_t, 415
alert, 416
msg_id, 416
prn, 416
raw_msg, 416
tow, 416
CNAV_OMEGA0_LSB
core_system_parameters, 160
CNAV_OMEGA_DOT_REF
core_system_parameters, 160
CNAV_OMEGA_LSB
core_system_parameters, 160
CNAV_PM_X_DOT_LSB
core_system_parameters, 160
CNAV_PM_X_LSB
core_system_parameters, 160
CNAV_PM_Y_DOT_LSB
core_system_parameters, 160
CNAV_PM_Y_LSB
core_system_parameters, 160
CNAV_QZSS_A_REF
core_system_parameters, 160
CNAV_T_EOP_LSB
core_system_parameters, 160

CNAV_TGD_LSB
core_system_parameters, 160
CNAV_TOC_LSB
core_system_parameters, 160
CNAV_TOE1_LSB
core_system_parameters, 160
CNAV_TOE2_LSB
core_system_parameters, 161
cnav_top
eph_t, 471
CNAV_TOP1_LSB
core_system_parameters, 161
CNAV_TOP2_LSB
core_system_parameters, 161
CNAV_TOT_LSB
core_system_parameters, 161
CNAV_TOW_LSB
core_system_parameters, 161
cnav_ura_valid
eph_t, 471
cnav_uraed
eph_t, 471
cnav_uraned0
eph_t, 471
cnav_uraned1
eph_t, 471
cnav_uraned2
eph_t, 472
cnav_v27_part_t, 416
crc_ok, 417
dec, 417
decisions, 417
decoded, 417
init, 417
invert, 417
message_lock, 417
n_crc_fail, 418
n_decoded, 418
n_symbols, 418
preamble_seen, 418
symbols, 418
CNAV_WN_LSF_LSB
core_system_parameters, 161
CNAV_WN_OT_LSB
core_system_parameters, 161
cnav_wnop
eph_t, 472
CnavSystem
core_system_parameters, 133
code
eph_t, 472
obsd_t, 789
pcv_t, 837
ssat_t, 981
code_bias
Galileo_HAS_data, 550
code_bias_flag
mt1_header, 755
code_bias_m
HAS_obs_corrections, 699
code_disc_sd_chips
Kf_Conf, 726
code_freq_chips
TrackingCmd, 1012
CODE_L1A
core_system_parameters, 161
CODE_L1B
core_system_parameters, 161
CODE_L1C
core_system_parameters, 161
CODE_L1D
core_system_parameters, 161
CODE_L1E
core_system_parameters, 162
CODE_L1I
core_system_parameters, 162
CODE_L1L
core_system_parameters, 162
CODE_L1M
core_system_parameters, 162
CODE_L1N
core_system_parameters, 162
CODE_L1P
core_system_parameters, 162
CODE_L1Q
core_system_parameters, 162
CODE_L1S
core_system_parameters, 162
CODE_L1W
core_system_parameters, 162
CODE_L1X
core_system_parameters, 162
CODE_L1Y
core_system_parameters, 163
CODE_L1Z
core_system_parameters, 163
CODE_L2C
core_system_parameters, 163
CODE_L2D
core_system_parameters, 163
CODE_L2I
core_system_parameters, 163
CODE_L2L
core_system_parameters, 163
CODE_L2M
core_system_parameters, 163
CODE_L2N
core_system_parameters, 163
CODE_L2P
core_system_parameters, 163
CODE_L2Q
core_system_parameters, 163
CODE_L2S
core_system_parameters, 164
CODE_L2W
core_system_parameters, 164

CODE_L2X
core_system_parameters, 164
CODE_L2Y
core_system_parameters, 164
CODE_L3I
core_system_parameters, 164
CODE_L3Q
core_system_parameters, 164
CODE_L3X
core_system_parameters, 164
CODE_L5A
core_system_parameters, 164
CODE_L5B
core_system_parameters, 164
CODE_L5C
core_system_parameters, 164
CODE_L5D
core_system_parameters, 165
CODE_L5I
core_system_parameters, 165
CODE_L5Q
core_system_parameters, 165
CODE_L5X
core_system_parameters, 165
CODE_L6A
core_system_parameters, 165
CODE_L6B
core_system_parameters, 165
CODE_L6C
core_system_parameters, 165
CODE_L6I
core_system_parameters, 165
CODE_L6L
core_system_parameters, 165
CODE_L6Q
core_system_parameters, 165
CODE_L6S
core_system_parameters, 166
CODE_L6X
core_system_parameters, 166
CODE_L6Z
core_system_parameters, 166
CODE_L7I
core_system_parameters, 166
CODE_L7Q
core_system_parameters, 166
CODE_L7X
core_system_parameters, 166
CODE_L8I
core_system_parameters, 166
CODE_L8Q
core_system_parameters, 166
CODE_L8X
core_system_parameters, 166
CODE_L9A
core_system_parameters, 166
CODE_L9B
core_system_parameters, 167
CODE_L9C
core_system_parameters, 167
CODE_L9X
core_system_parameters, 167
code_length
Acq_Conf, 333
Acq_Conf_Fpga, 337
code_length_chips
Acq_Conf_Fpga, 337
Dll_Pll_Conf_Fpga, 453
CODE_NONE
core_system_parameters, 167
code_on_L2
Gps_Ephemeris, 666
Code_Phase
Gps_Acq_Assist, 646
Code_Phase_int
Gps_Acq_Assist, 647
Code_phase_samples
Gnss_Synchro, 635
code_phase_sd_chips
Kf_Conf, 727
Code_Phase_window
Gps_Acq_Assist, 647
code_rate_cps
Acq_Conf_Fpga, 337
code_samples_per_chip
Dll_Pll_Conf_Fpga, 453
CODES_BDS
rtklib_rtcm3.h, 1236
CODES_GAL
rtklib_rtcm3.h, 1236
CODES_GLO
rtklib_rtcm3.h, 1237
CODES_GPS
rtklib_rtcm3.h, 1237
CODES_QZS
rtklib_rtcm3.h, 1237
CODES_SBS
rtklib_rtcm3.h, 1237
codesmooth
prcopt_t, 846
coef
lexion_t, 735
cog
Monitor_Pvt, 750
combined
Monitor_Pvt::TrackedSatelliteInfo, 1008
combined_view
Galileo_Ephemeris_Store, 539
Command_Event, 418
command_id, 418
event_type, 418
command_event.h, 1650, 1651
command_event_sptr
core_libs, 87
command_id
Command_Event, 418

comment
opt_t, 791
COMMENTH
algorithms_libs_rtklib, 271
common
Bds3_B1c_PageData, 364
complex_byte_to_float_x2, 419
complex_byte_to_float_x2.h, 1130
complex_byte_to_float_x2_sptr
algorithms_libs, 238
complex_exp_gen
algorithms_libs, 241
complex_exp_gen_conj
algorithms_libs, 241
complex_float_to_complex_byte, 419
complex_float_to_complex_byte.h, 1131
complex_float_to_complex_byte_sptr
algorithms_libs, 238
compressed_key
Osnma_Public_Key_Material, 804
compute_CMAC_AES
Gnss_Crypto, 608
compute_eph
Galileo_Reduced_CED, 565
compute_GLONASS_time
Glonass_Gnav_Ephemeris, 586
compute_grid
CudaPcpsEngine, 435
compute_HMAC_SHA_256
Gnss_Crypto, 608
compute_SHA3_256
Gnss_Crypto, 609
compute_SHA_256
Gnss_Crypto, 609
compute_visible_satellites
core_receiver, 90
computed_tag
Tag, 991
computeSamplesInFile
FileSourceBase, 493
Concurrent_Map< Data >, 420
get_map_copy, 420
read, 420
size, 420
write, 420
concurrent_map.h, 1697, 1698
Concurrent_Queue< Data >, 421
clear, 421
empty, 421
push, 421
size, 422
timed_wait_and_pop, 422
try_pop, 422
wait_and_pop, 422
concurrent_queue.h, 1699
conditioner_adapters, 228
Config
HistogramBitSynchronizer::Config, 423
configuration_interface.h, 1638
ConfigurationInterface, 425
configure_acquisition
Fpga_Acquisition, 508
conjugate_cc, 426
conjugate_cc.h, 1132, 1133
conjugate_cc_sptr
algorithms_libs, 238
conjugate_ic, 426
conjugate_ic.h, 1133, 1134
conjugate_ic_sptr
algorithms_libs, 238
conjugate_sc, 427
conjugate_sc.h, 1135
conjugate_sc_sptr
algorithms_libs, 238
connect
Ad936xCustomSignalSource, 346
Adrv9361z7035SignalSourceFPGA, 348
ArraySignalConditioner, 357
BeamformerFilter, 366
BladerfSignalSource, 401
ByteToShort, 402
Channel, 405
CshortToGrComplex, 432
CustomUDPSignalSource, 438
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
DMASignalSourceFPGA, 465
Evk1029SignalSource, 477
FifoSignalSource, 483
FileSourceBase, 493
FirFilter, 499
FlexibandSignalSource, 500
Fmcomms2SignalSource, 502
Fmcomms5SignalSourceFPGA, 504
FreqXlatingFirFilter, 518
GalileoE1TcpConnectorTracking, 572
GenSignalSource, 575
GnMaxSignalSource, 602
GNSSBlockInterface, 639
GNSSFlowgraph, 642
GpsL1CaDllPllTrackingGPU, 682
GpsL1CaGaussianTracking, 684
GpsL1CaKfTracking, 686
GpsL1CaTcpConnectorTracking, 689
GSS6450FileSignalSource, 695
HybridObservables, 706
IbyteToCbyte, 707
IbyteToComplex, 709
IbyteToCshort, 710
IONGSMSSignalSource, 720
IshortToComplex, 721
IshortToCshort, 723
LabsatSignalSource, 732
LimesdrSignalSource, 737
MAX2771EVKITSignalSourceFPGA, 746

MmseResamplerConditioner, 747
MultichannelFileSignalSource, 757
NotchFilter, 769
NotchFilterLite, 770
OsmosdrSignalSource, 793
Pass_Through, 806
PcpsAcquisitionAdapter, 829
PcpsAcquisitionAdapterCustom, 832
PcpsAcquisitionAdapterFpga, 835
PlutosdrSignalSource, 840
PocketSdrSignalSource, 842
PulseBlankingFilter, 853
RawArraySignalSource, 879
Rtklib_Pvt, 909
RtlTcpSignalSource, 928
SignalConditioner, 964
SignalGenerator, 966
TelemetryDecoderAdapter, 1000
UhdSignalSource, 1021
ZmqSignalSource, 1032
connected
GNSSFlowgraph, 642
Ntrip_Rtcm_Client_Status, 781
CONST_AMB
rtklib_ppp.h, 1226
consume_decoded_timestamps
sbas_l1_telemetry_decoder_gs, 933
context
cl_fft_plan, 412
continue_tag_accumulation
OSNMA_NavData, 801
control_thread.h, 1701
ControlThread, 427
~ControlThread, 428
applied_actions, 428
ControlThread, 428
flowgraph, 428
me, 429
processed_control_messages, 428
run, 429
set_control_queue, 429
cop
MACK_header, 738
MACK_tag_info, 741
Tag, 991
Core GNSS Receiver, 83
core_libs, 84
b_io_context, 86
channel_event_sptr, 86
channel_status_msg_receiver_sptr, 87
command_event_sptr, 87
find_uio_dev_file_name, 88
galileo_e6_has_msg_receiver_sptr, 87
GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER, 88
GALILEO_TOW_MAP_INVALID_TOW_MS, 88
GALILEO_TOW_MAP_INVALID_WEEK, 88
galileo_tow_map_sptr, 87
GALILEO_TOW_MAP_WEEK_MS, 88
GalileoTowMap, 87
GalileoTowMapMessage, 87
get_num_devices, 88
gnss_sdr_fpga_sample_counter_sptr, 87
gnss_sdr_sample_counter_sptr, 87
gnss_sdr_time_counter_sptr, 87
INI_ALLOW_MULTILINE, 86
ini_parse, 88
nav_message_monitor_sptr, 87
osnma_msg_receiver_sptr, 87
core_monitor, 89
b_io_context, 89
gnss_synchro_monitor_sptr, 89
core_receiver, 89
compute_visible_satellites, 90
core_system_parameters, 91
A0_6_LSB, 135
A1_6_LSB, 135
A_0_LSB, 135
A_0G_10_LSB, 135
A_1_LSB, 135
A_1_LSB_GAL, 135
A_1G_10_LSB, 135
A_F0_LSB, 136
A_F1_LSB, 136
A_F2_LSB, 136
AF0_10_LSB, 136
AF0_4_LSB, 136
AF0_8_LSB, 136
AF0_9_LSB, 136
AF1_10_LSB, 136
AF1_4_LSB, 136
AF1_8_LSB, 136
AF1_9_LSB, 136
AF2_4_LSB, 136
AI0_5_LSB, 137
AI1_5_LSB, 137
AI2_5_LSB, 137
ALM_AF0_LSB, 137
ALM_AF1_LSB, 137
ALM_DELTAI_LSB, 137
ALM_ECC_LSB, 137
ALM_MZERO_LSB, 137
ALM_OMEGA_LSB, 137
ALM_OMEGADOT_LSB, 137
ALM_OMEGAZERO_LSB, 137
ALM_SQUAREA_LSB, 137
ALM_TOA_LSB, 138
ALPHA_0, 133
ALPHA_0_LSB, 138
ALPHA_1_LSB, 138
ALPHA_2_LSB, 138
ALPHA_3_LSB, 138
AODO_LSB, 138
AS2R, 138
AU, 138
BDS_EPH_SOURCE_CNAV1, 138

BDS_EPH_SOURCE_CNAV2, 138
BEIDOU_B1C_CODE_LENGTH_CHIPS, 138
BEIDOU_B1C_CODE_PERIOD_MS, 139
BEIDOU_B1C_CODE_PERIOD_S, 139
BEIDOU_B1C_CODE_RATE_CPS, 139
BEIDOU_B1C_FRAME_LENGTH_SYMBOLS, 139
BEIDOU_B1C_FRAME_PERIOD_S, 139
BEIDOU_B1C_FREQ_HZ, 139
BEIDOU_B1C_NUMBER_OF_PRNS, 139
BEIDOU_B1C_OPT_ACQ_FS_SPS, 139
BEIDOU_B1C_PILOT_SECONDARY_CODE_LENGTH, 139
BEIDOU_B1C_SYMBOLS_PER_BIT, 139
BEIDOU_B1C_TELEMETRY_RATE_SPS, 140
BEIDOU_B1I_CODE_LENGTH_CHIPS, 140
BEIDOU_B1I_CODE_PERIOD_MS, 140
BEIDOU_B1I_CODE_PERIOD_S, 140
BEIDOU_B1I_CODE_RATE_CPS, 140
BEIDOU_B1I_D2_SECONDARY_CODE_STR, 140
BEIDOU_B1I_FREQ_HZ, 140
BEIDOU_B1I_GEO_PREAMBLE_LENGTH_SYMBOLS, 140
BEIDOU_B1I_GEO_PREAMBLE_SYMBOLS_STR, 140
BEIDOU_B1I_GEO_TELEMETRY_SYMBOLS_PER_BIT, 140
BEIDOU_B1I_OPT_ACQ_FS_SPS, 140
BEIDOU_B1I_PREAMBLE_DURATION_MS, 141
BEIDOU_B1I_PREAMBLE_DURATION_S, 141
BEIDOU_B1I_PREAMBLE_LENGTH_BITS, 141
BEIDOU_B1I_PREAMBLE_LENGTH_SYMBOLS, 141
BEIDOU_B1I_SECONDARY_CODE_LENGTH, 141
BEIDOU_B1I_SECONDARY_CODE_STR, 141
BEIDOU_B1I_TELEMETRY_RATE_BITS_SECOND, 141
BEIDOU_B1I_TELEMETRY_RATE_SYMBOLS_SECOND, 141
BEIDOU_B1I_TELEMETRY_SYMBOL_PERIOD_MS, 141
BEIDOU_B1I_TELEMETRY_SYMBOLS_PER_BIT, 141
BEIDOU_B2A_CODE_LENGTH_CHIPS, 141
BEIDOU_B2A_CODE_PERIOD_MS, 142
BEIDOU_B2A_CODE_PERIOD_S, 142
BEIDOU_B2A_CODE_RATE_CPS, 142
BEIDOU_B2A_FREQ_HZ, 142
BEIDOU_B2A_OPT_ACQ_FS_SPS, 142
BEIDOU_B2A_PREAMBLE_LENGTH_SYMBOLS, 142
BEIDOU_B2A_PREAMBLE_SYMBOLS_STR, 142
BEIDOU_B2A_SECONDARY_CODE_LENGTH, 142
BEIDOU_B2A_SECONDARY_CODE_STR, 142
BEIDOU_B2A_SYMBOLS_PER_BIT, 142
BEIDOU_B2A_TELEMETRY_RATE_SYMBOLS_SECOND, 142
BEIDOU_B3I_CODE_LENGTH_CHIPS, 143
BEIDOU_B3I_CODE_PERIOD_MS, 143
BEIDOU_B3I_CODE_PERIOD_S, 143
BEIDOU_B3I_CODE_RATE_CPS, 143
BEIDOU_B3I_D2_SECONDARY_CODE_STR, 143
BEIDOU_B3I_FREQ_HZ, 143
BEIDOU_B3I_GEO_PREAMBLE_LENGTH_SYMBOLS, 143
BEIDOU_B3I_GEO_PREAMBLE_SYMBOLS_STR, 143
BEIDOU_B3I_GEO_TELEMETRY_SYMBOLS_PER_BIT, 143
BEIDOU_B3I_OPT_ACQ_FS_SPS, 143
BEIDOU_B3I_PREAMBLE_DURATION_MS, 143
BEIDOU_B3I_PREAMBLE_DURATION_S, 144
BEIDOU_B3I_PREAMBLE_LENGTH_BITS, 144
BEIDOU_B3I_PREAMBLE_LENGTH_SYMBOLS, 144
BEIDOU_B3I_SECONDARY_CODE_LENGTH, 144
BEIDOU_B3I_SECONDARY_CODE_STR, 144
BEIDOU_B3I_TELEMETRY_RATE_BITS_SECOND, 144
BEIDOU_B3I_TELEMETRY_RATE_SYMBOLS_SECOND, 144
BEIDOU_B3I_TELEMETRY_SYMBOL_PERIOD_MS, 144
BEIDOU_B3I_TELEMETRY_SYMBOLS_PER_BIT, 144
beidou_bdgim_delay_m, 133
beidou_bdgim_nearest_odd_hour_mjd, 134
BEIDOU_CNAV1_A_DOT_LSB, 144
BEIDOU_CNAV1_A_REF_IGSO, 144
BEIDOU_CNAV1_A_REF_MEO, 145
BEIDOU_CNAV1_AF0_LSB, 145
BEIDOU_CNAV1_AF1_LSB, 145
BEIDOU_CNAV1_AF2_LSB, 145
BEIDOU_CNAV1_CIC_LSB, 145
BEIDOU_CNAV1_CIS_LSB, 145
BEIDOU_CNAV1_CRC_BITS, 145
BEIDOU_CNAV1_CRC_LSB, 145
BEIDOU_CNAV1_CRS_LSB, 145
BEIDOU_CNAV1_CUC_LSB, 145
BEIDOU_CNAV1_CUS_LSB, 145
BEIDOU_CNAV1_DELTA_A_LSB, 145
BEIDOU_CNAV1_DELTA_N0_DOT_LSB, 146
BEIDOU_CNAV1_DELTA_N0_LSB, 146
BEIDOU_CNAV1_E_LSB, 146
BEIDOU_CNAV1_FRAME_PERIOD_S, 146
BEIDOU_CNAV1_FRAME_SYMBOLS, 146
BEIDOU_CNAV1_H100_200_ELEMENT, 146
BEIDOU_CNAV1_H100_200_INDEX, 146
BEIDOU_CNAV1_H44_88_ELEMENT, 147
BEIDOU_CNAV1_H44_88_INDEX, 147
BEIDOU_CNAV1_I0_LSB, 147

BEIDOU_CNAV1_IDOT_LSB, 147
BEIDOU_CNAV1_INTERLEAVE_COLS, 147
BEIDOU_CNAV1_INTERLEAVE_ROWS, 147
BEIDOU_CNAV1_INTERLEAVED_SYMBOLS, 147
BEIDOU_CNAV1_ISC_LSB, 148
BEIDOU_CNAV1_LDPC_DC200, 148
BEIDOU_CNAV1_LDPC_DC88, 148
BEIDOU_CNAV1_LDPC_K200, 148
BEIDOU_CNAV1_LDPC_K88, 148
BEIDOU_CNAV1_LDPC_M200, 148
BEIDOU_CNAV1_LDPC_M88, 148
BEIDOU_CNAV1_LDPC_MAX_ITER, 148
BEIDOU_CNAV1_LDPC_N200, 148
BEIDOU_CNAV1_LDPC_N88, 148
BEIDOU_CNAV1_LDPC_NM, 148
BEIDOU_CNAV1_M0_LSB, 148
BEIDOU_CNAV1_OMEGA_LSB, 149
BEIDOU_CNAV1_OMEGADOT_LSB, 149
BEIDOU_CNAV1_PAGE_IONO_UTC, 149
BEIDOU_CNAV1_SF2_DATA_BITS, 149
BEIDOU_CNAV1_SF3_DATA_BITS, 149
BEIDOU_CNAV1_SOH_LSB_S, 149
BEIDOU_CNAV1_SUBFRAME1_SYMBOLS, 149
BEIDOU_CNAV1_SUBFRAME2_SYMBOLS, 149
BEIDOU_CNAV1_SUBFRAME3_SYMBOLS, 149
BEIDOU_CNAV1_SYMBOL_RATE_SPS, 149
BEIDOU_CNAV1_TGD_LSB, 149
BEIDOU_CNAV1_TOE_TOC_LSB, 149
BEIDOU_CNAV2_CRC_BITS, 150
BEIDOU_CNAV2_DATA_BITS, 150
BEIDOU_CNAV2_FRAME_MS, 150
BEIDOU_CNAV2_FRAME_PERIOD_S, 150
BEIDOU_CNAV2_FRAME_SYMBOLS, 150
BEIDOU_CNAV2_H48_96_ELEMENT, 150
BEIDOU_CNAV2_H48_96_INDEX, 150
BEIDOU_CNAV2_INFO_BITS, 150
BEIDOU_CNAV2_IODE_BITS, 151
BEIDOU_CNAV2_LDPC_DC, 151
BEIDOU_CNAV2_LDPC_K, 151
BEIDOU_CNAV2_LDPC_M, 151
BEIDOU_CNAV2_LDPC_N, 151
BEIDOU_CNAV2_LDPC_SYMBOLS, 151
BEIDOU_CNAV2_MES_TYPE_BITS, 151
BEIDOU_CNAV2_MS_PER_SYMBOL, 151
BEIDOU_CNAV2_MSG_ALM, 151
BEIDOU_CNAV2_MSG_CLK_ALM, 151
BEIDOU_CNAV2_MSG_CLK_DC, 151
BEIDOU_CNAV2_MSG_CLK_EOP, 151
BEIDOU_CNAV2_MSG_CLK_IONO, 152
BEIDOU_CNAV2_MSG_CLK_UTC, 152
BEIDOU_CNAV2_MSG_EPH1, 152
BEIDOU_CNAV2_MSG_EPH2, 152
BEIDOU_CNAV2_MT10_IODE_BIT, 152
BEIDOU_CNAV2_PREAMBLE_DETECTION_THRESHOLD, 152
BEIDOU_CNAV2_PREAMBLE_MS, 152
BEIDOU_CNAV2_PREAMBLE_SYMBOLS, 152
BEIDOU_CNAV2_SOW_LSB_S, 152
BEIDOU_CNAV2_SYMBOL_RATE_SPS, 152
BEIDOU_DNAV_BCH_CODE_BITS, 152
BEIDOU_DNAV_BCH_DATA_BITS, 152
BEIDOU_DNAV_BCH_PARITY_BITS, 153
BEIDOU_DNAV_BDT2GPST_LEAP_SEC_OFFSET, 153
BEIDOU_DNAV_BDT2GPST_WEEK_NUM_OFFSET, 153
BEIDOU_DNAV_D1_SUBFRAME_PERIOD_SECONDS, 153
BEIDOU_DNAV_D2_SUBFRAME1_PAGE_PERIOD_SECONDS, 153
BEIDOU_DNAV_PREAMBLE, 153
BEIDOU_DNAV_PREAMBLE_LENGTH_BITS, 153
BEIDOU_DNAV_PREAMBLE_LENGTH_SYMBOLS, 153
BEIDOU_DNAV_PREAMBLE_PERIOD_SYMBOLS, 153
BEIDOU_DNAV_SECONDS_PER_WEEK, 153
BEIDOU_DNAV_SUBFRAME_DATA_BITS, 153
BEIDOU_DNAV_SUBFRAME_SYMBOLS, 153
BEIDOU_DNAV_WORD_LENGTH_BITS, 154
BEIDOU_DNAV_WORDS_SUBFRAME, 154
BEIDOU_F, 154
BEIDOU_GM, 154
BEIDOU_OMEGA_EARTH_DOT, 154
BeidouCnav1LdpcGraph, 132
BETA_0_LSB, 154
BETA_1_LSB, 154
BETA_2_LSB, 154
BETA_3_LSB, 154
BGD_E1_E5A_5_LSB, 154
BGD_E1_E5B_5_LSB, 154
BITS_IN_OCTET, 155
C_IC_4_LSB, 155
C_IC_LSB, 155
C_IS_4_LSB, 155
C_IS_LSB, 155
C_RC_3_LSB, 155
C_RC_LSB, 155
C_RS_3_LSB, 155
C_RS_LSB, 155
C_UC_3_LSB, 155
C_UC_LSB, 155
C_US_3_LSB, 155
C_US_LSB, 156
CED_af0red_LSB, 156
CED_af1red_LSB, 156
CED_DeltaAred_LSB, 156
CED_Deltai0red_LSB, 156
CED_exred_LSB, 156
CED_eyred_LSB, 156
CED_lambda0red_LSB, 156
CED_Omega0red_LSB, 156
CNAV_A0_LSB, 156
CNAV_A1_LSB, 156

CNAV_A2_LSB, 156
CNAV_A_DOT_LSB, 157
CNAV_A_REF, 157
CNAV_AF0_LSB, 157
CNAV_AF1_LSB, 157
CNAV_AF2_LSB, 157
CNAV_ALPHA0_LSB, 157
CNAV_ALPHA1_LSB, 157
CNAV_ALPHA2_LSB, 157
CNAV_ALPHA3_LSB, 157
CNAV_BETA0_LSB, 157
CNAV_BETA1_LSB, 157
CNAV_BETA2_LSB, 157
CNAV_BETA3_LSB, 158
CNAV_CIC_LSB, 158
CNAV_CIS_LSB, 158
CNAV_CRC_LSB, 158
CNAV_CRS_LSB, 158
CNAV_CUC_LSB, 158
CNAV_CUS_LSB, 158
CNAV_DELTA_A_LSB, 158
CNAV_DELTA_N0_DOT_LSB, 158
CNAV_DELTA_N0_LSB, 158
CNAV_DELTA_OMEGA_DOT_LSB, 158
CNAV_DELTA_TLS_LSB, 158
CNAV_DELTA_TLSF_LSB, 159
CNAV_DELTA_UT1_GPS_DOT_LSB, 159
CNAV_DELTA_UT1_GPS_LSB, 159
CNAV_DN_LSB, 159
CNAV_E_ECCENTRICITY_LSB, 159
CNAV_I0_DOT_LSB, 159
CNAV_I0_LSB, 159
CNAV_ISCL1_LSB, 159
CNAV_ISCL2_LSB, 159
CNAV_ISCL5I_LSB, 159
CNAV_ISCL5Q_LSB, 159
CNAV_M0_LSB, 159
CNAV_OMEGA0_LSB, 160
CNAV_OMEGA_DOT_REF, 160
CNAV_OMEGA_LSB, 160
CNAV_PM_X_DOT_LSB, 160
CNAV_PM_X_LSB, 160
CNAV_PM_Y_DOT_LSB, 160
CNAV_PM_Y_LSB, 160
CNAV_QZSS_A_REF, 160
CNAV_T_EOP_LSB, 160
CNAV_TGD_LSB, 160
CNAV_TOC_LSB, 160
CNAV_TOE1_LSB, 160
CNAV_TOE2_LSB, 161
CNAV_TOP1_LSB, 161
CNAV_TOP2_LSB, 161
CNAV_TOT_LSB, 161
CNAV_TOW_LSB, 161
CNAV_WN_LSF_LSB, 161
CNAV_WN_OT_LSB, 161
CnavSystem, 133
CODE_L1A, 161
CODE_L1B, 161
CODE_L1C, 161
CODE_L1D, 161
CODE_L1E, 162
CODE_L1I, 162
CODE_L1L, 162
CODE_L1M, 162
CODE_L1N, 162
CODE_L1P, 162
CODE_L1Q, 162
CODE_L1S, 162
CODE_L1W, 162
CODE_L1X, 162
CODE_L1Y, 163
CODE_L1Z, 163
CODE_L2C, 163
CODE_L2D, 163
CODE_L2I, 163
CODE_L2L, 163
CODE_L2M, 163
CODE_L2N, 163
CODE_L2P, 163
CODE_L2Q, 163
CODE_L2S, 164
CODE_L2W, 164
CODE_L2X, 164
CODE_L2Y, 164
CODE_L3I, 164
CODE_L3Q, 164
CODE_L3X, 164
CODE_L5A, 164
CODE_L5B, 164
CODE_L5C, 164
CODE_L5D, 165
CODE_L5I, 165
CODE_L5Q, 165
CODE_L5X, 165
CODE_L6A, 165
CODE_L6B, 165
CODE_L6C, 165
CODE_L6I, 165
CODE_L6L, 165
CODE_L6Q, 165
CODE_L6S, 166
CODE_L6X, 166
CODE_L6Z, 166
CODE_L7I, 166
CODE_L7Q, 166
CODE_L7X, 166
CODE_L8I, 166
CODE_L8Q, 166
CODE_L8X, 166
CODE_L9A, 166
CODE_L9B, 167
CODE_L9C, 167
CODE_L9X, 167
CODE_NONE, 167
D1_A0_ALMANAC_LSB, 167

D1_A0_LSB, 167
D1_A0GAL_LSB, 167
D1_A0GLO_LSB, 167
D1_A0GPS_LSB, 167
D1_A0UTC_LSB, 167
D1_A1_ALMANAC_LSB, 167
D1_A1_LSB, 168
D1_A1GAL_LSB, 168
D1_A1GLO_LSB, 168
D1_A1GPS_LSB, 168
D1_A1UTC_LSB, 168
D1_A2_LSB, 168
D1_ALPHA0_LSB, 168
D1_ALPHA1_LSB, 168
D1_ALPHA2_LSB, 168
D1_ALPHA3_LSB, 168
D1_BETA0_LSB, 168
D1_BETA1_LSB, 168
D1_BETA2_LSB, 169
D1_BETA3_LSB, 169
D1_CIC_LSB, 169
D1_CIS_LSB, 169
D1_CRC_LSB, 169
D1_CRS_LSB, 169
D1_CUC_LSB, 169
D1_CUS_LSB, 169
D1_DELTA_I_LSB, 169
D1_DELTA_N_LSB, 169
D1_E_ALMANAC_LSB, 169
D1_E_LSB, 169
D1_I0_LSB, 170
D1_IDOT_LSB, 170
D1_M0_ALMANAC_LSB, 170
D1_M0_LSB, 170
D1_OMEGA0_ALMANAC_LSB, 170
D1_OMEGA0_LSB, 170
D1_OMEGA_ALMANAC_LSB, 170
D1_OMEGA_DOT_ALMANAC_LSB, 170
D1_OMEGA_DOT_LSB, 170
D1_OMEGA_LSB, 170
D1_SQRT_A_ALMANAC_LSB, 170
D1_SQRT_A_LSB, 170
D1_TGD1_LSB, 171
D1_TGD2_LSB, 171
D1_TOA_LSB, 171
D1_TOC_LSB, 171
D1_TOE_LSB, 171
D2_DELTA_T_LSB, 171
D2_ION_DELAY_LSB, 171
D2_ION_LOGICAL_FIRST_BIT, 171
D2_UDREI19_LOGICAL_FIRST_BIT, 171
D2_UDREI1_LOGICAL_FIRST_BIT, 171
D2R, 171
DELTA_A_7_LSB, 171
DELTA_A_8_LSB, 172
DELTA_A_9_LSB, 172
DELTA_I_7_LSB, 172
DELTA_I_8_LSB, 172
DELTA_I_9_LSB, 172
DELTA_N_3_LSB, 172
DELTA_N_LSB, 172
DELTAT_LS_LSB, 172
DELTAT_LSF_LSB, 172
DFRQ1_GLO, 172
DFRQ2_GLO, 172
DISPLAY_COLORS, 132
DN_LSB, 172
E_1_LSB, 173
E_7_LSB, 173
E_8_LSB, 173
E_9_LSB, 173
ECCENTRICITY_LSB, 173
FIRST_RS_BIT, 173
FIRST_RS_BIT_AFTER_IODNAV, 173
FNAV_A0_4_LSB, 173
FNAV_A0G_4_LSB, 173
FNAV_A12_2_LSB, 173
FNAV_A1_4_LSB, 173
FNAV_A1G_4_LSB, 173
FNAV_AF0_1_LSB, 174
FNAV_AF0_5_LSB, 174
FNAV_AF1_1_LSB, 174
FNAV_AF1_5_LSB, 174
FNAV_AF2_1_LSB, 174
FNAV_AI0_1_LSB, 174
FNAV_AI1_1_LSB, 174
FNAV_AI2_1_LSB, 174
FNAV_BGD_1_LSB, 174
FNAV_CIC_4_LSB, 174
FNAV_CIS_4_LSB, 174
FNAV_CRC_3_LSB, 174
FNAV_CRS_3_LSB, 175
FNAV_CUC_3_LSB, 175
FNAV_CUS_3_LSB, 175
FNAV_DELTAA12_5_LSB, 175
FNAV_DELTAI_5_LSB, 175
FNAV_DELTAN_3_LSB, 175
FNAV_E_2_LSB, 175
FNAV_E_5_LSB, 175
FNAV_I0_3_LSB, 175
FNAV_IDOT_2_LSB, 175
FNAV_M0_2_LSB, 175
FNAV_M0_5_LSB, 175
FNAV_OMEGA0_2_LSB, 176
FNAV_OMEGA0_5_LSB, 176
FNAV_OMEGADOT_2_LSB, 176
FNAV_OMEGADOT_5_LSB, 176
FNAV_T0A_5_LSB, 176
FNAV_T0C_1_LSB, 176
FNAV_T0E_3_LSB, 176
FNAV_T0G_4_LSB, 176
FNAV_T0T_4_LSB, 176
FNAV_W_3_LSB, 176
FNAV_W_5_LSB, 176
FREQ1, 176
FREQ1_BDS, 177

FREQ1_GLO, 177
FREQ2, 177
FREQ2_BDS, 177
FREQ2_GLO, 177
FREQ3_BDS, 177
FREQ3_GLO, 177
FREQ5, 177
FREQ6, 177
FREQ7, 177
FREQ8, 178
FREQ9, 178
GALILEO_CNAV_BITS_FOR_CRC, 178
GALILEO_CNAV_BYTES_FOR_CRC, 178
GALILEO_CNAV_CRC_LENGTH, 178
GALILEO_CNAV_HAS_PAGE_DATA_BITS, 178
GALILEO_CNAV_INFORMATION_VECTOR_LENGTH, 178
GALILEO_CNAV_INTERLEAVER_COLS, 178
GALILEO_CNAV_INTERLEAVER_ROWS, 178
GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK, 178
GALILEO_CNAV_MESSAGE_BITS_PER_PAGE, 178
GALILEO_CNAV_MT1_HEADER_BITS, 179
GALILEO_CNAV_OCTETS_IN_SUBPAGE, 179
GALILEO_CNAV_PAGE_HEADER_BITS, 179
GALILEO_CNAV_PAGE_MS, 179
GALILEO_CNAV_PAGE_RESERVED_BITS, 179
GALILEO_CNAV_PREAMBLE, 179
GALILEO_CNAV_PREAMBLE_LENGTH_BITS, 179
GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS, 179
GALILEO_CNAV_SYMBOLS_PER_PAGE, 179
GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND, 179
GALILEO_DATA_FRAME_BITS, 179
GALILEO_DATA_FRAME_BYTES, 179
GALILEO_DATA_JK_BITS, 180
GALILEO_E1_B_CODE_LENGTH_CHIPS, 180
GALILEO_E1_B_PRIMARY_CODE, 180
GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH, 180
GALILEO_E1_B_SAMPLES_PER_SYMBOL, 180
GALILEO_E1_B_SYMBOL_RATE_BPS, 180
GALILEO_E1_C_PRIMARY_CODE, 180
GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH, 180
GALILEO_E1_C_SECONDARY_CODE, 180
GALILEO_E1_C_SECONDARY_CODE_LENGTH, 180
GALILEO_E1_CODE_CHIP_RATE_CPS, 180
GALILEO_E1_CODE_PERIOD_MS, 181
GALILEO_E1_CODE_PERIOD_S, 181
GALILEO_E1_FREQ_HZ, 181
GALILEO_E1_HISTORY_DEEP, 181
GALILEO_E1_NUMBER_OF_CODES, 181
GALILEO_E1_OPT_ACQ_FS_SPS, 181
GALILEO_E1_SUB_CARRIER_A_RATE_HZ, 181
GALILEO_E1_SUB_CARRIER_B_RATE_HZ, 181
GALILEO_E5A_CODE_CHIP_RATE_CPS, 181
GALILEO_E5A_CODE_LENGTH_CHIPS, 181
GALILEO_E5A_CODE_PERIOD_MS, 182
GALILEO_E5A_CODE_PERIOD_S, 182
GALILEO_E5A_CRC_ERROR_LIMIT, 182
GALILEO_E5A_FREQ_HZ, 182
GALILEO_E5A_HISTORY_DEEP, 182
GALILEO_E5A_I_PRIMARY_CODE, 182
GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH, 182
GALILEO_E5A_I_SECONDARY_CODE, 182
GALILEO_E5A_I_SECONDARY_CODE_LENGTH, 182
GALILEO_E5A_I_TIERED_CODE_PERIOD_S, 182
GALILEO_E5A_NUMBER_OF_CODES, 182
GALILEO_E5A_OPT_ACQ_FS_SPS, 183
GALILEO_E5A_Q_PRIMARY_CODE, 183
GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH, 183
GALILEO_E5A_Q_SECONDARY_CODE, 183
GALILEO_E5A_Q_SECONDARY_CODE_LENGTH, 183
GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH, 183
GALILEO_E5A_Q_TIERED_CODE_PERIOD_S, 183
GALILEO_E5A_SYMBOL_RATE_BPS, 183
GALILEO_E5B_CODE_CHIP_RATE_CPS, 183
GALILEO_E5B_CODE_LENGTH_CHIPS, 183
GALILEO_E5B_CODE_PERIOD_MS, 183
GALILEO_E5B_CODE_PERIOD_S, 184
GALILEO_E5B_FREQ_HZ, 184
GALILEO_E5B_HISTORY_DEEP, 184
GALILEO_E5B_I_PRIMARY_CODE, 184
GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH, 184
GALILEO_E5B_I_SECONDARY_CODE, 184
GALILEO_E5B_I_SECONDARY_CODE_LENGTH, 184
GALILEO_E5B_I_TIERED_CODE_PERIOD_S, 184
GALILEO_E5B_NUMBER_OF_CODES, 184
GALILEO_E5B_OPT_ACQ_FS_SPS, 184
GALILEO_E5B_Q_PRIMARY_CODE, 184
GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH, 185
GALILEO_E5B_Q_SECONDARY_CODE, 185
GALILEO_E5B_Q_SECONDARY_CODE_LENGTH, 185
GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH, 185
GALILEO_E5B_Q_TIERED_CODE_PERIOD_S, 185
GALILEO_E5B_SYMBOL_RATE_BPS, 185
GALILEO_E6_B_CODE_CHIP_RATE_CPS, 185

GALILEO_E6_B_CODE_LENGTH_CHIPS, 185
GALILEO_E6_B_PRIMARY_CODE, 185
GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH, 185
GALILEO_E6_C_CODE_CHIP_RATE_CPS, 185
GALILEO_E6_C_CODE_LENGTH_CHIPS, 186
GALILEO_E6_C_PRIMARY_CODE, 186
GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH, 186
GALILEO_E6_C_SECONDARY_CODE, 186
GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS, 186
GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH, 186
GALILEO_E6_CODE_PERIOD_MS, 186
GALILEO_E6_CODE_PERIOD_S, 186
GALILEO_E6_FREQ_HZ, 186
GALILEO_E6_NUMBER_OF_CODES, 186
GALILEO_E6_OPT_ACQ_FS_SPS, 186
GALILEO_F, 187
GALILEO_FNAV_CODES_PER_PAGE, 187
GALILEO_FNAV_CODES_PER_PREAMBLE, 187
GALILEO_FNAV_CODES_PER_SYMBOL, 187
GALILEO_FNAV_DATA_FRAME_BITS, 187
GALILEO_FNAV_DATA_FRAME_BYTES, 187
GALILEO_FNAV_INTERLEAVER_COLS, 187
GALILEO_FNAV_INTERLEAVER_ROWS, 187
GALILEO_FNAV_PAGE_TYPE_BITS, 187
GALILEO_FNAV_PREAMBLE, 187
GALILEO_FNAV_PREAMBLE_LENGTH_BITS, 187
GALILEO_FNAV_SECONDS_PER_PAGE, 187
GALILEO_FNAV_SYMBOLS_PER_PAGE, 188
GALILEO_GM, 188
GALILEO_HAS_HALF_WEEK_SECONDS, 188
GALILEO_HAS_INVALID_TOW, 188
GALILEO_HAS_INVALID_WEEK, 188
GALILEO_HAS_SECONDS_PER_HOUR, 188
GALILEO_HAS_SECONDS_PER_WEEK, 188
GALILEO_INAV_DUMMY_WORD_TYPE, 188
GALILEO_INAV_ENCODED_SSP1, 188
GALILEO_INAV_ENCODED_SSP2, 188
GALILEO_INAV_ENCODED_SSP3, 188
GALILEO_INAV_EVEN_PAGE_TYPE_BIT, 188
GALILEO_INAV_INTERLEAVER_COLS, 189
GALILEO_INAV_INTERLEAVER_ROWS, 189
GALILEO_INAV_ODD_PAGE_TYPE_BIT, 189
GALILEO_INAV_PAGE_PART_MS, 189
GALILEO_INAV_PAGE_PART_SECONDS, 189
GALILEO_INAV_PAGE_PART_SYMBOLS, 189
GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS, 189
GALILEO_INAV_PAGE_SECONDS, 189
GALILEO_INAV_PAGE_SYMBOLS, 189
GALILEO_INAV_PLAIN_SSP1, 189
GALILEO_INAV_PLAIN_SSP2, 189
GALILEO_INAV_PLAIN_SSP3, 190
GALILEO_INAV_PREAMBLE, 190
GALILEO_INAV_PREAMBLE_LENGTH_BITS, 190
GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS, 190
GALILEO_ISM_CRC_DATA_BITS, 190
GALILEO_ISM_CRC_DATA_BYTES, 190
Galileo_Nav_Message_Source, 133
Galileo_Nav_Message_Type, 133
GALILEO_PAGE_TYPE_BITS, 190
GALILEO_TELEMETRY_RATE_BITS_SECOND, 190
GLONASS_C20, 190
GLONASS_CA_NBR_SATS, 190
GLONASS_EARTH_INCLINATION, 190
GLONASS_EARTH_RADIUS, 190
GLONASS_F_M_A, 191
GLONASS_FLATTENING, 191
GLONASS_GM, 191
GLONASS_GNAV_CRC_I_INDEX, 191
GLONASS_GNAV_CRC_J_INDEX, 191
GLONASS_GNAV_CRC_K_INDEX, 191
GLONASS_GNAV_CRC_L_INDEX, 191
GLONASS_GNAV_CRC_M_INDEX, 191
GLONASS_GNAV_CRC_N_INDEX, 191
GLONASS_GNAV_CRC_P_INDEX, 192
GLONASS_GNAV_CRC_Q_INDEX, 192
GLONASS_GNAV_DATA_SYMBOLS, 192
GLONASS_GNAV_ECC_LOCATOR, 192
GLONASS_GNAV_HAMMING_CODE_BITS, 192
GLONASS_GNAV_PREAMBLE, 132
GLONASS_GNAV_PREAMBLE_DURATION_S, 192
GLONASS_GNAV_PREAMBLE_LENGTH_BITS, 192
GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS, 192
GLONASS_GNAV_PREAMBLE_PERIOD_BITS, 192
GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS, 192
GLONASS_GNAV_PREAMBLE_SAMPLES, 132
GLONASS_GNAV_PREAMBLE_STR, 193
GLONASS_GNAV_SECONDS_PER_WEEK, 193
GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE, 193
GLONASS_GNAV_STRING_BITS, 193
GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE, 193
GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND, 193
GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND, 193
GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT, 193
GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BI 193
GLONASS_GRAVITY, 193
GLONASS_GRAVITY_CORRECTION, 193

GLONASS_J2, 194
GLONASS_J4, 194
GLONASS_J6, 194
GLONASS_J8, 194
GLONASS_L1_CA_CHIP_PERIOD_S, 194
GLONASS_L1_CA_CODE_LENGTH_CHIPS, 194
GLONASS_L1_CA_CODE_PERIOD_MS, 194
GLONASS_L1_CA_CODE_PERIOD_S, 194
GLONASS_L1_CA_CODE_RATE_CPS, 194
GLONASS_L1_CA_DFREQ_HZ, 194
GLONASS_L1_CA_FREQ_HZ, 195
GLONASS_L1_CA_HISTORY_DEEP, 195
GLONASS_L1_CA_SYMBOL_PERIOD_S, 195
GLONASS_L1_CA_SYMBOL_RATE_BPS, 195
GLONASS_L2_CA_CHIP_PERIOD_S, 195
GLONASS_L2_CA_CODE_LENGTH_CHIPS, 195
GLONASS_L2_CA_CODE_PERIOD_MS, 195
GLONASS_L2_CA_CODE_PERIOD_S, 195
GLONASS_L2_CA_CODE_RATE_CPS, 195
GLONASS_L2_CA_DFREQ_HZ, 195
GLONASS_L2_CA_FREQ_HZ, 196
GLONASS_L2_CA_SYMBOL_PERIOD_S, 196
GLONASS_L2_CA_SYMBOL_RATE_BPS, 196
GLONASS_LEAP_SECONDS, 196
GLONASS_MOON_ECCENTRICITY, 196
GLONASS_MOON_GM, 196
GLONASS_MOON_INCLINATION, 196
GLONASS_MOON_OMEGA_0, 197
GLONASS_MOON_OMEGA_1, 197
GLONASS_MOON_Q0, 197
GLONASS_MOON_Q1, 197
GLONASS_MOON_SEMI_MAJOR_AXIS, 197
GLONASS_OMEGA_EARTH_DOT, 197
GLONASS_PRN, 197
GLONASS_SEMI_MAJOR_AXIS, 197
GLONASS_SUN_ECCENTRICITY, 197
GLONASS_SUN_GM, 197
GLONASS_SUN_OMEGA, 198
GLONASS_SUN_Q0, 198
GLONASS_SUN_Q1, 198
GLONASS_SUN_SEMI_MAJOR_AXIS, 198
GLONASS_TAU_0, 198
GLONASS_TAU_1, 198
GLONASS_U0, 198
GNSS_OMEGA_EARTH_DOT, 198
GNSS_PI, 198
GPS_CA_PREAMBLE, 198
GPS_CA_PREAMBLE_DURATION_MS, 199
GPS_CA_PREAMBLE_DURATION_S, 199
GPS_CA_PREAMBLE_LENGTH_BITS, 199
GPS_CA_PREAMBLE_LENGTH_SYMBOLS, 199
GPS_CA_PREAMBLE_SYMBOLS_STR, 199
GPS_CA_TELEMETRY_RATE_BITS_SECOND, 199
GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND, 199
GPS_CA_TELEMETRY_SYMBOLS_PER_BIT, 199
GPS_CNAV_DATA_PAGE_BITS, 199
GPS_F, 199
GPS_GM, 199
GPS_L1_CA_BIT_PERIOD_MS, 200
GPS_L1_CA_CHIP_PERIOD_S, 200
GPS_L1_CA_CODE_LENGTH_CHIPS, 200
GPS_L1_CA_CODE_PERIOD_MS, 200
GPS_L1_CA_CODE_PERIOD_S, 200
GPS_L1_CA_CODE_RATE_CPS, 200
GPS_L1_CA_HISTORY_DEEP, 200
GPS_L1_CA_OPT_ACQ_FS_SPS, 200
GPS_L1_FREQ_HZ, 200
GPS_L2_CNAV_DATA_PAGE_BITS, 200
GPS_L2_CNAV_DATA_PAGE_DURATION_S, 201
GPS_L2_CNAV_DATA_PAGE_SYMBOLS, 201
GPS_L2_FREQ_HZ, 201
GPS_L2_L_CODE_LENGTH_CHIPS, 201
GPS_L2_L_CODE_RATE_CPS, 201
GPS_L2_L_PERIOD_S, 201
GPS_L2_M_CODE_LENGTH_CHIPS, 201
GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED, 201
GPS_L2_M_CODE_PERIOD_MS, 201
GPS_L2_M_CODE_RATE_CPS, 201
GPS_L2_M_CODE_RATE_L_ZEROED_CPS, 202
GPS_L2_M_PERIOD_S, 202
GPS_L2_SAMPLES_PER_SYMBOL, 202
GPS_L2_SYMBOLS_PER_BIT, 202
GPS_L2C_HISTORY_DEEP, 202
GPS_L2C_M_INIT_REG, 202
GPS_L2C_OPT_ACQ_FS_SPS, 202
GPS_L5_CNAV_DATA_PAGE_BITS, 203
GPS_L5_CNAV_DATA_PAGE_DURATION_S, 203
GPS_L5_CNAV_DATA_PAGE_SYMBOLS, 203
GPS_L5_FREQ_HZ, 203
GPS_L5_HISTORY_DEEP, 203
GPS_L5_OPT_ACQ_FS_SPS, 203
GPS_L5_SAMPLES_PER_SYMBOL, 203
GPS_L5_SYMBOLS_PER_BIT, 203
GPS_L5I_CODE_LENGTH_CHIPS, 203
GPS_L5I_CODE_RATE_CPS, 203
GPS_L5I_INIT_REG, 203
GPS_L5I_NH_CODE, 204
GPS_L5I_NH_CODE_LENGTH, 204
GPS_L5I_NH_CODE_STR, 204
GPS_L5I_PERIOD_MS, 204
GPS_L5I_PERIOD_S, 204
GPS_L5I_SYMBOL_PERIOD_MS, 204
GPS_L5I_SYMBOL_PERIOD_S, 204
GPS_L5Q_CODE_LENGTH_CHIPS, 204
GPS_L5Q_CODE_RATE_CPS, 204
GPS_L5Q_INIT_REG, 204
GPS_L5Q_NH_CODE, 204
GPS_L5Q_NH_CODE_LENGTH, 205
GPS_L5Q_NH_CODE_STR, 205
GPS_L5Q_PERIOD_S, 205
GPS_LNAV_DATA_ID, 205
gps_ref_week_from_config, 134

GPS_SUBFRAME_BITS, 205
GPS_SUBFRAME_LENGTH, 205
GPS_SUBFRAME_MS, 205
GPS_SUBFRAME_SECONDS, 205
GPS_SV_CONFIG_AS_FLAG_MASK, 205
GPS_SV_CONFIG_BLOCK_IIF, 205
GPS_SV_CONFIG_BLOCK_IIR_M, 206
GPS_SV_CONFIG_CODE_MASK, 206
GPS_SV_CONFIG_UNKNOWN, 206
GPS_WORD_BITS, 206
GPS_WORD_LENGTH, 206
HALF_PI, 206
HAS_MSG_CODE_BIAS_LENGTH, 206
HAS_MSG_CODE_BIAS_SCALE_FACTOR, 206
HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH, 206
HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH, 206
HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH, 206
HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH, 207
HAS_MSG_DELTA_CLOCK_SCALE_FACTOR, 207
HAS_MSG_DELTA_CROSS_TRACK_LENGTH, 207
HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR, 207
HAS_MSG_DELTA_IN_TRACK_LENGTH, 207
HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR, 207
HAS_MSG_DELTA_RADIAL_LENGTH, 207
HAS_MSG_DELTA_RADIAL_SCALE_FACTOR, 207
HAS_MSG_GALILEO_SYSTEM, 207
HAS_MSG_GPS_SYSTEM, 207
HAS_MSG_ID_CLOCK_SUBSET_LENGTH, 207
HAS_MSG_ID_MASK_LENGTH, 207
HAS_MSG_IOD_GAL_LENGTH, 208
HAS_MSG_IOD_GPS_LENGTH, 208
HAS_MSG_MASK_RESERVED_LENGTH, 208
HAS_MSG_NAV_MESSAGE_LENGTH, 208
HAS_MSG_NSYS_LENGTH, 208
HAS_MSG_NSYS_SUB_LENGTH, 208
HAS_MSG_NUMBER_GNSS_IDS, 208
HAS_MSG_NUMBER_MASK_IDS, 208
HAS_MSG_NUMBER_MAX_TOH, 208
HAS_MSG_NUMBER_MESSAGE_IDS, 208
HAS_MSG_NUMBER_SATELLITE_IDS, 208
HAS_MSG_NUMBER_SIGNAL_MASKS, 208
HAS_MSG_PHASE_BIAS_LENGTH, 209
HAS_MSG_PHASE_BIAS_SCALE_FACTOR, 209
HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH, 209
HAS_MSG_SATELLITE_MASK_LENGTH, 209
HAS_MSG_SIGNAL_MASK_LENGTH, 209
HAS_MSG_VALIDITY_INDEX_LENGTH, 209
HAS_MSG_WRONG_SYSTEM, 209
HAS_SIGNAL_INDEX_TABLE, 209
HAS_VALIDITY_INTERVALS, 209
I_0_2_LSB, 210
I_0_LSB, 210
I_DOT_2_LSB, 210
I_DOT_LSB, 210
INAV_RS_BUFFER_LENGTH, 210
INAV_RS_INFO_VECTOR_LENGTH, 210
INAV_RS_PARITY_VECTOR_LENGTH, 210
INAV_RS_SUBVECTOR_LENGTH, 210
LnavSystem, 133
M0_10_LSB, 210
M0_1_LSB, 210
M0_7_LSB, 210
M0_9_LSB, 210
M_0_LSB, 211
MAX_SECONDS_REMEMBERING_MASK_IOD, 211
MAX_SECONDS_REMEMBERING_MID, 211
MAX_TOA_DELAY_MS, 211
MAX_CODE, 211
OMEGA0_10_LSB, 211
OMEGA0_7_LSB, 211
OMEGA0_8_LSB, 211
OMEGA_0_2_LSB, 211
OMEGA_0_LSB, 211
OMEGA_2_LSB, 211
OMEGA_7_LSB, 212
OMEGA_8_LSB, 212
OMEGA_9_LSB, 212
OMEGA_DOT_10_LSB, 212
OMEGA_DOT_3_LSB, 212
OMEGA_DOT_7_LSB, 212
OMEGA_DOT_8_LSB, 212
OMEGA_DOT_LSB, 212
OMEGA_LSB, 212
OSNMA_TABLE_1, 212
OSNMA_TABLE_10, 212
OSNMA_TABLE_11, 213
OSNMA_TABLE_15, 213
OSNMA_TABLE_16, 213
OSNMA_TABLE_2, 213
OSNMA_TABLE_3, 214
OSNMA_TABLE_5, 214
OSNMA_TABLE_6, 214
OSNMA_TABLE_7, 214
OSNMA_TABLE_8, 215
parse_agnss_ref_location, 134
parse_agnss_ref_utc_time, 134
PI_TWO_N19, 215
PI_TWO_N23, 215
PI_TWO_N31, 215
PI_TWO_N38, 215
PI_TWO_N43, 215
QZSS_CA_PREAMBLE_SYMBOLS_STR, 215
qzss_l1cb_prn_to_nominal_prn, 135
QZSS_L5I_NH_CODE_STR, 215
QZSS_L5Q_NH_CODE_STR, 216

R2D, 216
SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS, 216
SBAS_L1_CHANNEL_SYMBOL_RATE_SPS, 216
SBAS_L1_CODE_LENGTH_CHIPS, 216
SBAS_L1_CODE_PERIOD_MS, 216
SBAS_L1_CODE_PERIOD_S, 216
SBAS_L1_CODE_RATE_CPS, 216
SBAS_L1_DATA_FIELD_BYTES, 216
SBAS_L1_DATA_RATE_BPS, 216
SBAS_L1_FREQ_HZ, 217
SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS, 217
SBAS_L1_MSG_LENGTH_BITS, 217
SBAS_L1_MSG_LENGTH_BYTES, 217
SBAS_L1_OPT_ACQ_FS_SPS, 217
SBAS_L1_PREAMBLE_1, 217
SBAS_L1_PREAMBLE_2, 217
SBAS_L1_PREAMBLE_3, 217
SBAS_L1_PRN_MAX, 217
SBAS_L1_PRN_MIN, 217
SC2RAD, 218
SIGNAL_FREQ_MAP, 218
SIZE_DSM_BLOCKS_BYTES, 218
SPEED_OF_LIGHT_M_MS, 218
SPEED_OF_LIGHT_M_S, 218
SQRT_A_LSB, 218
T0A_7_LSB, 218
T0A_9_LSB, 218
T0C_4_LSB, 219
T0E_1_LSB, 219
T0T_6_LSB, 219
T_0_G_10_LSB, 219
T_GD_LSB, 219
T_OA, 135
T_OA_LSB, 219
T_OC_LSB, 219
T_OE_LSB, 219
T_OT_LSB, 219
TEXT_BLACK, 219
TEXT_BLUE, 219
TEXT_BOLD_BLACK, 219
TEXT_BOLD_BLUE, 220
TEXT_BOLD_CYAN, 220
TEXT_BOLD_GREEN, 220
TEXT_BOLD_MAGENTA, 220
TEXT_BOLD_RED, 220
TEXT_BOLD_WHITE, 220
TEXT_BOLD_YELLOW, 220
TEXT_CYAN, 220
TEXT_GREEN, 220
TEXT_MAGENTA, 220
TEXT_RED, 220
TEXT_RESET, 220
TEXT_WHITE, 221
TEXT_YELLOW, 221
TWO_N10, 221
TWO_N11, 221
TWO_N14, 221
TWO_N15, 221
TWO_N16, 221
TWO_N17, 221
TWO_N18, 221
TWO_N19, 221
TWO_N2, 222
TWO_N20, 222
TWO_N21, 222
TWO_N22, 222
TWO_N23, 222
TWO_N24, 222
TWO_N25, 222
TWO_N26, 222
TWO_N27, 222
TWO_N29, 222
TWO_N30, 223
TWO_N31, 223
TWO_N32, 223
TWO_N33, 223
TWO_N34, 223
TWO_N35, 223
TWO_N38, 223
TWO_N39, 223
TWO_N40, 223
TWO_N43, 223
TWO_N44, 224
TWO_N46, 224
TWO_N48, 224
TWO_N5, 224
TWO_N50, 224
TWO_N51, 224
TWO_N55, 224
TWO_N57, 224
TWO_N59, 224
TWO_N6, 224
TWO_N60, 225
TWO_N66, 225
TWO_N68, 225
TWO_N8, 225
TWO_N9, 225
TWO_P11, 225
TWO_P12, 225
TWO_P14, 225
TWO_P16, 225
TWO_P19, 225
TWO_P3, 226
TWO_P31, 226
TWO_P32, 226
TWO_P4, 226
TWO_P56, 226
TWO_P57, 226
TWO_P8, 226
TWO_PI, 226
WN_LSF_LSB, 226
WN_T_LSB, 226
corr
half_cyc_tag, 696

corrections
Beidou_Dnav_Differential_Corrections, 382
correlation_length_ms
Gnss_Synchro, 635
CosBocCorrelationFunction
tracking_libs, 320
counter
MACK_tag_and_info, 741
countwords_
Allocator, 352
cov
peph_t, 838
cov_xx
Monitor_Pvt, 750
cov_xy
Monitor_Pvt, 750
cov_yy
Monitor_Pvt, 750
cov_yz
Monitor_Pvt, 750
cov_zx
Monitor_Pvt, 750
cov_zz
Monitor_Pvt, 750
cp
rtcm_t, 902
cpks
DSM_nma_header, 468
Cpu_Multicorrelator, 429
cpu_multicorrelator.h, 1604
Cpu_Multicorrelator_16sc, 430
cpu_multicorrelator_16sc.h, 1605, 1606
Cpu_Multicorrelator_Real_Codes, 430
cpu_multicorrelator_real_codes.h, 1606, 1607
cputime
rtksvr_t, 923
Crc
Gnss_Ephemeris, 614
crc
eph_t, 472
crc_ok
cnav_v27_part_t, 417
CRC_test
Glonass_Gnav_Navigation_Message, 594
create_file_source_hook
FileTimestampSignalSource, 497
FourBitCpxFileSignalSource, 506
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
SpirFileSignalSource, 980
TwoBitCpxFileSignalSource, 1016
TwoBitPackedFileSignalSource, 1019
createProtobuffer
Serdes_Galileo_Eph, 953
Serdes_Gnss_Synchro, 954
Serdes_Gps_Eph, 956
Serdes_Monitor_Pvt, 957
Serdes_Nav_Message, 959
cross_track_m
HAS_orbit_corrections, 700
Crs
Gnss_Ephemeris, 614
crs
eph_t, 472
cshort_to_float_x2, 431
cshort_to_float_x2.h, 1136, 1137
cshort_to_float_x2_sptr
algorithms_libs, 238
cshort_to_gr_complex, 431
cshort_to_gr_complex.h, 1102, 1103
cshort_to_gr_complex_sptr
data_type_gr_blocks, 229
cshort_to_grcomplex.h, 1093
CshortToGrComplex, 432
connect, 432
disconnect, 432
get_left_block, 433
get_right_block, 433
implementation, 433
item_size, 433
role, 433
CTR
Tag, 992
CubatureFilter, 433
Cuc
Gnss_Ephemeris, 614
cuc
eph_t, 472
CUDA_CALLABLE_MEMBER_DEVICE
cuda_multicorrelator.h, 1608
CUDA_CALLABLE_MEMBER_GLOBAL
cuda_multicorrelator.h, 1608
cuda_device
Acq_Conf, 333
cuda_grid_count
pcps_acquisition, 809
cuda_multicorrelator, 434
cuda_multicorrelator.h, 1608, 1609
CUDA_CALLABLE_MEMBER_DEVICE, 1608
CUDA_CALLABLE_MEMBER_GLOBAL, 1608
cuda_pcps_engine.h, 1078
cuda_ready
pcps_acquisition, 809
CudaPcpsEngine, 434
compute_grid, 435
CudaPcpsEngine, 435
device_bytes, 436
device_name, 436
GridId, 435
is_valid, 436
last_error, 436
MAIN_GRID, 435
set_doppler_wipeoffs, 436
set_fft_codes, 436
STEP2_GRID, 435
Cus

Gnss_Ephemeris, 614
cus
eph_t, 472
custom_udp_signal_source.h, 1401, 1402
CustomUDPSignalSource, 437
connect, 438
disconnect, 438
get_left_block, 438
get_right_block, 438
item_size, 438
cycle
rtksvr_t, 923
cyclic
solbuf_t, 973

D
obsd_t, 789
D1_A0_ALMANAC_LSB
core_system_parameters, 167
D1_A0_LSB
core_system_parameters, 167
D1_A0GAL_LSB
core_system_parameters, 167
D1_A0GLO_LSB
core_system_parameters, 167
D1_A0GPS_LSB
core_system_parameters, 167
D1_A0UTC_LSB
core_system_parameters, 167
D1_A1_ALMANAC_LSB
core_system_parameters, 167
D1_A1_LSB
core_system_parameters, 168
D1_A1GAL_LSB
core_system_parameters, 168
D1_A1GLO_LSB
core_system_parameters, 168
D1_A1GPS_LSB
core_system_parameters, 168
D1_A1UTC_LSB
core_system_parameters, 168
D1_A2_LSB
core_system_parameters, 168
D1_ALPHA0_LSB
core_system_parameters, 168
D1_ALPHA1_LSB
core_system_parameters, 168
D1_ALPHA2_LSB
core_system_parameters, 168
D1_ALPHA3_LSB
core_system_parameters, 168
D1_BETA0_LSB
core_system_parameters, 168
D1_BETA1_LSB
core_system_parameters, 168
D1_BETA2_LSB
core_system_parameters, 169
D1_BETA3_LSB
core_system_parameters, 169
D1_CIC_LSB
core_system_parameters, 169
D1_CIS_LSB
core_system_parameters, 169
D1_CRC_LSB
core_system_parameters, 169
D1_CRS_LSB
core_system_parameters, 169
D1_CUC_LSB
core_system_parameters, 169
D1_CUS_LSB
core_system_parameters, 169
D1_DELTA_I_LSB
core_system_parameters, 169
D1_DELTA_N_LSB
core_system_parameters, 169
D1_E_ALMANAC_LSB
core_system_parameters, 169
D1_E_LSB
core_system_parameters, 169
D1_I0_LSB
core_system_parameters, 170
D1_IDOT_LSB
core_system_parameters, 170
D1_M0_ALMANAC_LSB
core_system_parameters, 170
D1_M0_LSB
core_system_parameters, 170
D1_OMEGA0_ALMANAC_LSB
core_system_parameters, 170
D1_OMEGA0_LSB
core_system_parameters, 170
D1_OMEGA_ALMANAC_LSB
core_system_parameters, 170
D1_OMEGA_DOT_ALMANAC_LSB
core_system_parameters, 170
D1_OMEGA_DOT_LSB
core_system_parameters, 170
D1_OMEGA_LSB
core_system_parameters, 170
D1_SQRT_A_ALMANAC_LSB
core_system_parameters, 170
D1_SQRT_A_LSB
core_system_parameters, 170
d1_subframe_decoder
Beidou_Dnav_Navigation_Message, 392
D1_TGD1_LSB
core_system_parameters, 171
D1_TGD2_LSB
core_system_parameters, 171
D1_TOA_LSB
core_system_parameters, 171
D1_TOC_LSB
core_system_parameters, 171
D1_TOE_LSB
core_system_parameters, 171
D2_DELTA_T_LSB
core_system_parameters, 171

D2_ION_DELAY_LSB
core_system_parameters, 171
D2_ION_LOGICAL_FIRST_BIT
core_system_parameters, 171
d2_subframe_decoder
Beidou_Dnav_Navigation_Message, 392
D2_UDREI19_LOGICAL_FIRST_BIT
core_system_parameters, 171
D2_UDREI1_LOGICAL_FIRST_BIT
core_system_parameters, 171
D2R
core_system_parameters, 171
d_acc
Sbas_Ephemeris, 930
d_af0
Sbas_Ephemeris, 931
d_af1
Sbas_Ephemeris, 931
d_AXn
Glonass_Gnav_Ephemeris, 588
d_AYn
Glonass_Gnav_Ephemeris, 588
d_AZn
Glonass_Gnav_Ephemeris, 588
d_B1
Glonass_Gnav_Utc_Model, 600
d_B2
Glonass_Gnav_Utc_Model, 600
d_B_n
Glonass_Gnav_Ephemeris, 588
d_C_n
Glonass_Gnav_Almanac, 581
d_Delta_i_n_A
Glonass_Gnav_Almanac, 581
d_Delta_T_n_A
Glonass_Gnav_Almanac, 581
d_Delta_T_n_A_dot
Glonass_Gnav_Almanac, 582
d_Delta_tau_n
Glonass_Gnav_Ephemeris, 588
d_dsm_header
OSNMA_data, 794
d_dsm_kroot_message
OSNMA_data, 794
d_dsm_kroot_new_message
OSNMA_data, 794
d_dsm_pkr_message
OSNMA_data, 794
d_dtr
Glonass_Gnav_Ephemeris, 589
d_E_n
Glonass_Gnav_Ephemeris, 589
d_epsilon_n_A
Glonass_Gnav_Almanac, 582
d_F_T
Glonass_Gnav_Ephemeris, 589
d_gamma_n
Glonass_Gnav_Ephemeris, 589
d_H_n_A
Glonass_Gnav_Almanac, 582
d_iode
Glonass_Gnav_Ephemeris, 589
d_KP
Glonass_Gnav_Almanac, 582
Glonass_Gnav_Utc_Model, 600
d_kp_leap_correction_s
Glonass_Gnav_Ephemeris, 589
d_l3rd_n
Glonass_Gnav_Ephemeris, 589
d_l5th_n
Glonass_Gnav_Ephemeris, 589
d_l_n
Glonass_Gnav_Almanac, 582
d_lambda_n_A
Glonass_Gnav_Almanac, 582
d_M
Glonass_Gnav_Ephemeris, 590
d_m
Glonass_Gnav_Ephemeris, 590
d_M_n_A
Glonass_Gnav_Almanac, 582
d_mack_message
OSNMA_data, 794
d_n
Glonass_Gnav_Ephemeris, 590
d_N_4
Glonass_Gnav_Almanac, 582
Glonass_Gnav_Utc_Model, 600
d_N_A
Glonass_Gnav_Almanac, 583
Glonass_Gnav_Utc_Model, 600
d_n_A
Glonass_Gnav_Almanac, 583
d_N_T
Glonass_Gnav_Ephemeris, 590
d_nav_data
OSNMA_data, 795
d_nma_header
OSNMA_data, 795
d_omega_n_A
Glonass_Gnav_Almanac, 583
d_P
Glonass_Gnav_Ephemeris, 590
d_P_1
Glonass_Gnav_Ephemeris, 590
d_P_2
Glonass_Gnav_Ephemeris, 590
d_P_3
Glonass_Gnav_Ephemeris, 590
d_P_4
Glonass_Gnav_Ephemeris, 591
d_pos
Sbas_Ephemeris, 931
d_satClkDrift
Glonass_Gnav_Ephemeris, 591
d_t_b

Glonass_Gnav_Ephemeris, 591
d_t_k
Glonass_Gnav_Ephemeris, 591
d_t_lambda_n_A
Glonass_Gnav_Almanac, 583
d_tau_c
Glonass_Gnav_Ephemeris, 591
Glonass_Gnav_Utc_Model, 601
d_tau_gps
Glonass_Gnav_Utc_Model, 601
d_tau_n
Glonass_Gnav_Ephemeris, 591
d_tau_n_A
Glonass_Gnav_Almanac, 583
d_tod
Glonass_Gnav_Ephemeris, 591
d_tof
Sbas_Ephemeris, 931
d_TOW
Glonass_Gnav_Ephemeris, 591
d_vel
Sbas_Ephemeris, 931
d_VXn
Glonass_Gnav_Ephemeris, 592
d_VYn
Glonass_Gnav_Ephemeris, 592
d_VZn
Glonass_Gnav_Ephemeris, 592
d_WN
Glonass_Gnav_Ephemeris, 592
d_Xn
Glonass_Gnav_Ephemeris, 592
d_Yn
Glonass_Gnav_Ephemeris, 592
d_yr
Glonass_Gnav_Ephemeris, 592
d_Zn
Glonass_Gnav_Ephemeris, 592
daf0
sbslcorr_t, 939
daf1
sbslcorr_t, 939
data
an_packet_t, 356
erp_t, 475
obs_t, 788
solbuf_t, 973
solstatbuf_t, 977
tec_t, 997
tle_t, 1002
Data Type Adapters, 228
data_
IONGSMSChunkUnpackingCtx< WT >, 717
data_codes
Dll_Pll_Conf_Fpga, 453
data_filepath
IONGSMSFileSource::SegmentDescriptor, 945
data_type
matlab_type_traits< double >, 742
matlab_type_traits< float >, 742
matlab_type_traits< int32_t >, 743
matlab_type_traits< int64_t >, 743
matlab_type_traits< uint32_t >, 744
matlab_type_traits< uint64_t >, 744
matlab_type_traits< uint8_t >, 745
data_type_adapter
ArraySignalConditioner, 357
SignalConditioner, 964
data_type_adapters, 228
data_type_gr_blocks, 229
cshort_to_gr_complex_sptr, 229
interleaved_byte_to_complex_byte_sptr, 229
interleaved_byte_to_complex_short_sptr, 229
interleaved_short_to_complex_short_sptr, 229
datum
solopt_t, 974
dclk
ssr_t, 983
ddeph
ssr_t, 983
dec
cnav_v27_part_t, 417
decisions
cnav_v27_part_t, 417
v27_t, 1028
decisions_count
v27_t, 1028
decisions_index
v27_t, 1028
decode
Geohash, 577
ReedSolomon, 881
Viterbi_Decoder, 1029
decode_block
Viterbi_Decoder_Sbas, 1030
decode_item_type
SignalSourceBase, 968
decoded
cnav_v27_part_t, 417
decoder_errors
Ntrip_Rtcm_Client_Status, 781
DEFAULT_CARRIER_SMOOTHING_FACTOR
gnss_sdr_flags, 259
degf
solopt_t, 974
del
sta_t, 985
delay
sbsigp_t, 937
Deleter, 438
buffer_, 439
Deleter, 439
operator(), 439
deln
eph_t, 472
delta_A

Gps_CNAV_Ephemeris, 655
DELTA_A_7
Galileo_Almanac_Helper, 524
DELTA_A_7_LSB
core_system_parameters, 171
DELTA_A_8
Galileo_Almanac_Helper, 524
DELTA_A_8_LSB
core_system_parameters, 172
DELTA_A_9
Galileo_Almanac_Helper, 525
DELTA_A_9_LSB
core_system_parameters, 172
delta_a_m
Bds3_B1c_AlmanacReduced, 360
delta_clock_correction
Galileo_HAS_data, 550
delta_clock_correction_clock_subset
Galileo_HAS_data, 550
delta_clock_multiplier
Galileo_HAS_data, 550
delta_clock_multiplier_clock_subset
Galileo_HAS_data, 550
delta_cross_track
Galileo_HAS_data, 550
delta_i
Gnss_Almanac, 605
delta_i_7
Galileo_Almanac_Helper, 525
DELTA_I_7_LSB
core_system_parameters, 172
delta_i_8
Galileo_Almanac_Helper, 525
DELTA_I_8_LSB
core_system_parameters, 172
delta_i_9
Galileo_Almanac_Helper, 525
DELTA_I_9_LSB
core_system_parameters, 172
delta_i_rad
Bds3_B1c_AlmanacMedium, 359
delta_in_track
Galileo_HAS_data, 550
delta_n
Gnss_Ephemeris, 615
DELTA_N_3_LSB
core_system_parameters, 172
DELTA_N_LSB
core_system_parameters, 172
delta_ndot
Gnss_Ephemeris, 615
delta_OMEGAdot
Gps_CNAV_Ephemeris, 656
delta_radial
Galileo_HAS_data, 551
delta_t
Beidou_Dnav_Differential_Correction, 381
delta_t_available
Beidou_Dnav_Differential_Correction, 381
delta_t_LS
Beidou_Cnav1_Utc_Model, 376
delta_t_LSF
Beidou_Cnav1_Utc_Model, 376
Delta_tLS
Galileo_Utc_Model, 571
Delta_tLSF
Galileo_Utc_Model, 571
delta_ut1_dot_s_day
Bds3_B1c_Eop, 362
delta_ut1_gps
Gps_CNAV_Eop, 651
delta_ut1_gps_dot
Gps_CNAV_Eop, 651
delta_ut1_s
Bds3_B1c_Eop, 362
DeltaAred
Galileo_Reduced_CED, 565
Deltai0red
Galileo_Reduced_CED, 565
DeltaT_LS
Beidou_Dnav_Utc_Model, 397
Gps_Utc_Model, 680
DELTAT_LS_LSB
core_system_parameters, 172
DeltaT_LSF
Beidou_Dnav_Utc_Model, 397
Gps_Utc_Model, 681
DELTAT_LSF_LSB
core_system_parameters, 172
deltype
sta_t, 986
deph
ssr_t, 984
desig
tled_t, 1003
dev
serial_t, 960
dev_t
algorithms_libs_rtklib, 271
device_bytes
CudaPcpsEngine, 436
device_name
Acq_Conf_Fpga, 337
CudaPcpsEngine, 436
Dll_Pll_Conf_Fpga, 454
DFRQ1_GLO
core_system_parameters, 172
DFRQ2_GLO
core_system_parameters, 172
dgps
nav_t, 762
rtcm_t, 902
dgps_t, 439
iod, 439
prc, 439
rrc, 439

t0, 439
udre, 440
dif
Bds3_B1c_PageCommon, 363
dim
cl_fft_plan, 412
dir
kernel_info_t, 724
url_t, 1026
direct_resampler_conditioner.h, 1383
direct_resampler_conditioner_cb, 440
sample_freq_in, 440
sample_freq_out, 440
direct_resampler_conditioner_cb.h, 1386
direct_resampler_conditioner_cb_sptr
resampler_gr_blocks, 304
direct_resampler_conditioner_cc, 441
sample_freq_in, 441
sample_freq_out, 441
direct_resampler_conditioner_cc.h, 1388
direct_resampler_conditioner_cc_sptr
resampler_gr_blocks, 304
direct_resampler_conditioner_cs, 441
sample_freq_in, 442
sample_freq_out, 442
direct_resampler_conditioner_cs.h, 1389, 1390
direct_resampler_conditioner_cs_sptr
resampler_gr_blocks, 304
DirectResamplerConditioner, 442
connect, 443
disconnect, 443
get_left_block, 443
get_right_block, 443
implementation, 443
item_size, 443
role, 443
disable_secondary_codes
Fpga_Multicorrelator_8sc, 514
disconnect
Ad936xCustomSignalSource, 346
Adrv9361z7035SignalSourceFPGA, 348
ArraySignalConditioner, 357
BeamformerFilter, 366
BladerfSignalSource, 401
ByteToShort, 402
Channel, 405
CshortToGrComplex, 432
CustomUDPSignalSource, 438
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
DMASignalSourceFPGA, 465
Evk1029SignalSource, 477
FifoSignalSource, 483
FileSourceBase, 493
FirFilter, 499
FlexibandSignalSource, 500
Fmcomms2SignalSource, 502
Fmcomms5SignalSourceFPGA, 504
FreqXlatingFirFilter, 518
GalileoE1TcpConnectorTracking, 572
GenSignalSource, 575
GnMaxSignalSource, 602
GNSSBlockInterface, 640
GNSSFlowgraph, 642
GpsL1CaDllPllTrackingGPU, 682
GpsL1CaGaussianTracking, 684
GpsL1CaKfTracking, 686
GpsL1CaTcpConnectorTracking, 689
GSS6450FileSignalSource, 695
HybridObservables, 706
IbyteToCbyte, 707
IbyteToComplex, 709
IbyteToCshort, 710
IONGSMSSignalSource, 720
IshortToComplex, 721
IshortToCshort, 723
LabsatSignalSource, 732
LimesdrSignalSource, 737
MAX2771EVKITSignalSourceFPGA, 746
MmseResamplerConditioner, 747
MultichannelFileSignalSource, 757
NotchFilter, 769
NotchFilterLite, 770
OsmosdrSignalSource, 793
Pass_Through, 806
PcpsAcquisitionAdapter, 829
PcpsAcquisitionAdapterCustom, 832
PcpsAcquisitionAdapterFpga, 835
PlutosdrSignalSource, 840
PocketSdrSignalSource, 842
PulseBlankingFilter, 853
RawArraySignalSource, 879
Rtklib_Pvt, 909
RtlTcpSignalSource, 928
SignalConditioner, 965
SignalGenerator, 966
TelemetryDecoderAdapter, 1000
UhdSignalSource, 1021
ZmqSignalSource, 1032
display.h, 1779, 1780
DISPLAY_COLORS
core_system_parameters, 132
display_rate_ms
Pvt_Conf, 856
dll_bw_hz
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
dll_bw_narrow_hz
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
dll_filter_order
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
dll_nc_e_minus_l_normalized
tracking_libs, 320

dll_nc_vemlp_normalized
tracking_libs, 321
Dll_Pll_Conf, 444
b1c_data_prompt_scale, 445
b1c_pilot_prompt_scale, 445
b1c_prompt_normalize_power, 445
b1c_prompt_use_data_q, 445
b1c_secondary_lock_ratio, 445
bit_synchronization_time_limit_s, 445
bs_dominance_ratio, 445
bs_min_events_for_lock, 445
bs_min_prompt_mag, 446
bs_runner_up_margin, 446
bs_stable_best_required, 446
bs_tentative_events_required, 446
bs_transition_confidence, 446
bs_transition_window_epochs, 446
bs_use_phase_dot_detector, 446
carrier_aiding, 446
carrier_lock_test_smoother_alpha, 446
carrier_lock_test_smoother_samples, 446
carrier_lock_th, 446
cn0_min, 446
cn0_samples, 447
cn0_smoother_alpha, 447
cn0_smoother_samples, 447
dll_bw_hz, 447
dll_bw_narrow_hz, 447
dll_filter_order, 447
dll_pull_in_bw_hz, 447
dump, 447
dump_filename, 447
dump_mat, 447
early_late_space_chips, 447
early_late_space_narrow_chips, 447
enable_doppler_correction, 448
enable_fll_pull_in, 448
enable_fll_steady_state, 448
extend_correlation_symbols, 448
f_error_accumulation, 448
f_error_doppler_step, 448
f_error_dump, 448
f_error_dump_filename, 448
f_error_step_num, 448
fll_bw_hz, 448
fll_filter_order, 448
fs_in, 448
high_dyn, 449
item_type, 449
max_carrier_lock_fail, 449
max_code_lock_fail, 449
pll_bw_hz, 449
pll_bw_narrow_hz, 449
pll_filter_order, 449
pll_pull_in_bw_hz, 449
pull_in_time_s, 449
qmboc, 449
signal, 449
slope, 449
smoother_length, 450
spc, 450
system, 450
tow_to_trk, 450
track_pilot, 450
vector_length, 450
very_early_late_space_chips, 450
very_early_late_space_narrow_chips, 450
y_intercept, 450
dll_pll_conf.h, 1611
Dll_Pll_Conf_Fpga, 450
bit_synchronization_time_limit_s, 452
bs_dominance_ratio, 452
bs_min_events_for_lock, 452
bs_min_prompt_mag, 452
bs_runner_up_margin, 452
bs_stable_best_required, 452
bs_tentative_events_required, 452
bs_transition_confidence, 452
bs_transition_window_epochs, 452
bs_use_phase_dot_detector, 452
ca_codes, 453
carrier_aiding, 453
carrier_lock_test_smoother_alpha, 453
carrier_lock_test_smoother_samples, 453
carrier_lock_th, 453
cn0_min, 453
cn0_samples, 453
cn0_smoother_alpha, 453
cn0_smoother_samples, 453
code_length_chips, 453
code_samples_per_chip, 453
data_codes, 453
device_name, 454
dll_bw_hz, 454
dll_bw_narrow_hz, 454
dll_filter_order, 454
dll_pull_in_bw_hz, 454
dump, 454
dump_filename, 454
dump_mat, 454
early_late_space_chips, 454
early_late_space_narrow_chips, 454
enable_doppler_correction, 454
enable_fll_pull_in, 454
enable_fll_steady_state, 455
extend_correlation_symbols, 455
extend_fpga_integration_periods, 455
extended_correlation_in_fpga, 455
fll_bw_hz, 455
fll_filter_order, 455
fpga_integration_period, 455
fs_in, 455
high_dyn, 455
max_carrier_lock_fail, 455
max_code_lock_fail, 455
pll_bw_hz, 455

pll_bw_narrow_hz, 456
pll_filter_order, 456
pll_pull_in_bw_hz, 456
pull_in_time_s, 456
signal, 456
slope, 456
smoother_length, 456
spc, 456
system, 456
track_pilot, 456
vector_length, 456
very_early_late_space_chips, 456
very_early_late_space_narrow_chips, 457
y_intercept, 457
dll_pll_conf_fpga.h, 1612, 1613
dll_pll_tracking_adapter.h, 1563
dll_pll_tracking_adapter_fpga.h, 1565
dll_pll_veml_tracking, 457
DllPllTrackingFrequencyErrorTest, 458
f_error_bin_multiplier, 458
dll_pll_veml_tracking.h, 1575, 1576
dll_pll_veml_tracking_fpga, 458
~dll_pll_veml_tracking_fpga, 459
general_work, 459
reset, 459
set_channel, 459
set_gnss_synchro, 459
start_tracking, 459
stop_tracking, 459
dll_pll_veml_tracking_fpga.h, 1579, 1580
dll_pll_veml_tracking_fpga_sptr
tracking_gr_blocks, 316
dll_pll_veml_tracking_sptr
tracking_gr_blocks, 316
dll_pull_in_bw_hz
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
DllPllTrackingAdapter, 460
~DllPllTrackingAdapter, 461
connect, 461
disconnect, 461
DllPllTrackingAdapter, 461
get_left_block, 461
get_right_block, 461
implementation, 461
item_size, 461
role, 462
set_channel, 462
set_gnss_synchro, 462
start_tracking, 462
stop_tracking, 462
DllPllTrackingAdapterFpga, 462
connect, 463
disconnect, 463
get_left_block, 463
get_right_block, 463
implementation, 463
item_size, 463
role, 464
set_channel, 464
set_gnss_synchro, 464
start_tracking, 464
stop_tracking, 464
DllPllTrackingFrequencyErrorTest
dll_pll_veml_tracking, 458
DMA_close
Fpga_DMA, 511
DMA_open
Fpga_DMA, 511
dma_signal_source_fpga.h, 1403, 1404
DMA_write
Fpga_DMA, 511
DMASignalSourceFPGA, 464
connect, 465
disconnect, 465
get_left_block, 465
get_right_block, 465
item_size, 465
start, 466
DN
Beidou_Cnav1_Utc_Model, 376
Beidou_Dnav_Utc_Model, 397
Galileo_Utc_Model, 571
Gps_Utc_Model, 681
DN_LSB
core_system_parameters, 172
dominance_ratio
HistogramBitSynchronizer::Config, 423
Doppler0
Gps_Acq_Assist, 647
Doppler1
Gps_Acq_Assist, 647
doppler_max
Acq_Conf, 333
Acq_Conf_Fpga, 337
doppler_min
Acq_Conf, 333
doppler_step
Acq_Conf, 333
Acq_Conf_Fpga, 337
doppler_step2
Acq_Conf, 333
Acq_Conf_Fpga, 337
dopplerUncertainty
Gps_Acq_Assist, 647
downsampling_factor
Acq_Conf_Fpga, 337
downsampling_filter_delay
Acq_Conf_Fpga, 337
downsampling_filter_num
Acq_Conf_Fpga, 338
dpCA
raw_t, 875
dpos
sbslcorr_t, 939
ds

DSM_KROOT_message, 467
dsm_block_id
DSM_dsm_header, 466
DSM_dsm_header, 466
dsm_block_id, 466
dsm_id, 466
dsm_id
DSM_dsm_header, 466
DSM_KROOT_message, 466
alpha, 467
cidkr, 467
ds, 467
hf, 467
kroot, 467
ks, 467
maclt, 467
mf, 467
nb_dk, 467
p_dk, 467
pkid, 467
reserved, 468
reserved1, 468
towh_k, 468
ts, 468
verified, 468
wn_k, 468
DSM_nma_header, 468
cid, 468
cpks, 468
nmas, 469
reserved, 469
DSM_PKR_message, 469
itn, 469
mid, 469
nb_dp, 469
npk, 469
npkt, 469
npktid, 469
p_dp, 470
dt
sbsfcorr_t, 936
dtaun
geph_t, 578
dtr
Gnss_Ephemeris, 615
sol_t, 971
DTTOL
algorithms_libs_rtklib, 279
dump
Acq_Conf, 333
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
Obs_Conf, 787
Pvt_Conf, 856
Tlm_Conf, 1004
dump_channel
Acq_Conf, 333
dump_crc_stats
Tlm_Conf, 1004
dump_crc_stats_filename
Tlm_Conf, 1005
dump_filename
Acq_Conf, 334
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
Obs_Conf, 787
Pvt_Conf, 856
Tlm_Conf, 1005
dump_logger_helper.h, 1137, 1138
write_value, 1138
dump_mat
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
Obs_Conf, 787
Pvt_Conf, 856
Tlm_Conf, 1005
dvel
sbslcorr_t, 939
dynamics
prcopt_t, 846

e
alm_t, 352
eph_t, 472
E1B_DVS
Galileo_Ephemeris, 537
Galileo_Reduced_CED, 566
E1B_HS
Galileo_Almanac, 522
Galileo_Ephemeris, 537
Galileo_Reduced_CED, 566
E1B_HS_10
Galileo_Almanac_Helper, 525
E1B_HS_8
Galileo_Almanac_Helper, 525
E1B_HS_9
Galileo_Almanac_Helper, 525
E5a_DVS
Galileo_Ephemeris, 537
E5a_HS
Galileo_Almanac, 522
Galileo_Ephemeris, 537
E5a_HS_10
Galileo_Almanac_Helper, 525
E5a_HS_8
Galileo_Almanac_Helper, 525
E5a_HS_9
Galileo_Almanac_Helper, 525
E5b_DVS
Galileo_Ephemeris, 537
Galileo_Reduced_CED, 566
E5b_HS
Galileo_Almanac, 522
Galileo_Ephemeris, 537

Galileo_Reduced_CED, 566
E5b_HS_10
Galileo_Almanac_Helper, 525
E5b_HS_8
Galileo_Almanac_Helper, 525
E5b_HS_9
Galileo_Almanac_Helper, 526
E_1_LSB
core_system_parameters, 173
e_7
Galileo_Almanac_Helper, 526
E_7_LSB
core_system_parameters, 173
e_8
Galileo_Almanac_Helper, 526
E_8_LSB
core_system_parameters, 173
e_9
Galileo_Almanac_Helper, 526
E_9_LSB
core_system_parameters, 173
e_verification_status
Tag, 991
early_late_space_chips
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
early_late_space_narrow_chips
Dll_Pll_Conf, 447
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
ecc
Gnss_Almanac, 605
Gnss_Ephemeris, 615
tled_t, 1003
eccentricity
Bds3_B1c_AlmanacMedium, 359
ECCENTRICITY_LSB
core_system_parameters, 173
edc.h, 1549
edge_phase
HistogramBitSynchronizer, 702
EFACT_BDS
algorithms_libs_rtklib, 279
EFACT_GAL
algorithms_libs_rtklib, 279
EFACT_GLO
algorithms_libs_rtklib, 280
EFACT_GPS
algorithms_libs_rtklib, 280
EFACT_IRN
algorithms_libs_rtklib, 280
EFACT_QZS
algorithms_libs_rtklib, 280
EFACT_SBS
algorithms_libs_rtklib, 280
el
solstat_t, 976
eleno
tled_t, 1003
Elevation
Gps_Acq_Assist, 647
elevation_deg
Monitor_Pvt::TrackedSatelliteInfo, 1008
elmaskar
prcopt_t, 846
elmaskhold
prcopt_t, 846
elmin
prcopt_t, 846
empty
Concurrent_Queue< Data >, 421
Galileo_Ephemeris_Store, 539
Secure_String, 944
ena
exterr_t, 480
snrmask_t, 971
ENABDS
algorithms_libs_rtklib, 271
enable_carrier_nco_cmd
TrackingCmd, 1012
enable_carrier_smoothing
Obs_Conf, 788
enable_code_nco_cmd
TrackingCmd, 1012
enable_doppler_correction
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 454
Kf_Conf, 727
enable_E6
Obs_Conf, 788
enable_fll_pull_in
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 454
enable_fll_steady_state
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 455
enable_monitor
Obs_Conf, 788
enable_monitor_output
Acq_Conf, 334
enable_navdata_monitor
Tlm_Conf, 1005
enable_pvt_kf
Pvt_Conf, 856
enable_reed_solomon
Galileo_Inav_Message, 556
Tlm_Conf, 1005
enable_rx_clock_correction
Pvt_Conf, 856
enable_secondary_codes
Fpga_Multicorrelator_8sc, 514
enable_zero_padding
Acq_Conf_Fpga, 338
enabled
Ntrip_Rtcm_Client_Config, 779

SatelliteVisibility, 929
ENAGLO
algorithms_libs_rtklib, 271
ENAQZS
algorithms_libs_rtklib, 271
encode
Geohash, 577
encode_with_generator_matrix
ReedSolomon, 881
encode_with_generator_poly
ReedSolomon, 881
encoding_from_string
ion_gsms_stream_encodings.h, 1510
end
solbuf_t, 973
eop
Bds3_B1c_PageData, 364
eph
nav_t, 762
eph_t, 470
A, 471
Adot, 471
apply_has_corrections, 471
cic, 471
cis, 471
cnav_top, 471
cnav_ura_valid, 471
cnav_uraed, 471
cnav_uraned0, 471
cnav_uraned1, 471
cnav_uraned2, 472
cnav_wnop, 472
code, 472
crc, 472
crs, 472
cuc, 472
cus, 472
deln, 472
e, 472
f0, 472
f1, 472
f2, 472
fit, 473
flag, 473
has_clock_correction_m, 473
has_orbit_cross_track_correction_m, 473
has_orbit_in_track_correction_m, 473
has_orbit_radial_correction_m, 473
i0, 473
idot, 473
iodc, 473
iode, 473
isc, 473
M0, 473
ndot, 474
omg, 474
OMG0, 474
OMGd, 474
sat, 474
sva, 474
svh, 474
tgd, 474
toc, 474
toe, 474
toes, 474
ttr, 474
week, 475
eph_to_rtklib
algorithms_libs_rtklib, 277
ephemerides
Ntrip_Rtcm_Snapshot, 782
Ephemeris_Map
Galileo_Ephemeris_Store, 539
EPHOPT_BRDC
algorithms_libs_rtklib, 280
EPHOPT_LEX
algorithms_libs_rtklib, 280
EPHOPT_PREC
algorithms_libs_rtklib, 280
EPHOPT_SBAS
algorithms_libs_rtklib, 280
EPHOPT_SSRAPC
algorithms_libs_rtklib, 281
EPHOPT_SSRCOM
algorithms_libs_rtklib, 281
ephsat
raw_t, 875
rtcm_t, 902
epoch
ambc_t, 354
tled_t, 1003
epoch_ms
HistogramBitSynchronizer::Config, 423
epochs_until_next_edge
HistogramBitSynchronizer, 702
eratio
prcopt_t, 846
erp
nav_t, 762
erp_t, 475
data, 475
n, 475
nmax, 475
erpd_t, 475
lod, 476
mjd, 476
ut1_utc, 476
xp, 476
xpr, 476
yp, 476
ypr, 476
err
prcopt_t, 847
ERR_BRDCI
algorithms_libs_rtklib, 281
ERR_CBIAS

algorithms_libs_rtklib, 281
ERR_ION
rtklib_pntpos.h, 1221
ERR_SAAS
algorithms_libs_rtklib, 281
ERR_TROP
rtklib_pntpos.h, 1221
errbuf
rtk_t, 905
error
ftp_t, 519
serial_t, 960
estqzsisb
prcopt_t, 847
etype
tled_t, 1003
event_type
Channel_Event, 407
Command_Event, 418
evk1029_signal_source.h, 1405, 1406
evk1029_source.h, 1462, 1463
Evk1029SignalSource, 476
connect, 477
disconnect, 477
get_right_block, 477
item_size, 477
Evk1029Source, 478
Evk1029Source_sptr
signal_source_gr_blocks, 308
excludelimit
Acq_Conf_Fpga, 338
excsat
rtk_t, 905
expanded
Beidou_Dnav_Almanac, 380
expanded_corrections
Beidou_Dnav_Differential_Corrections, 382
Exponential_Smoother, 478
∼Exponential_Smoother, 479
Exponential_Smoother, 479
operator=, 479
set_alpha, 479
set_samples_for_initialization, 479
exponential_smoother.h, 1614, 1615
exred
Galileo_Reduced_CED, 566
exsats
prcopt_t, 847
extend_correlation_symbols
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 455
Kf_Conf, 727
extend_fpga_integration_periods
Dll_Pll_Conf_Fpga, 455
extended_correlation_in_fpga
Dll_Pll_Conf_Fpga, 455
exterr
prcopt_t, 847
EXTERR_CLK
rtklib_preceph.h, 1230
EXTERR_EPH
rtklib_preceph.h, 1230
exterr_t, 480
cerr, 480
ena, 480
gloicb, 480
gpsglob, 480
perr, 480
eyred
Galileo_Reduced_CED, 566

f0
alm_t, 352
eph_t, 472
f1
alm_t, 352
eph_t, 472
f2
eph_t, 472
f_error_accumulation
Dll_Pll_Conf, 448
f_error_bin_multiplier
dll_pll_veml_tracking, 458
f_error_doppler_step
Dll_Pll_Conf, 448
f_error_dump
Dll_Pll_Conf, 448
f_error_dump_filename
Dll_Pll_Conf, 448
f_error_step_num
Dll_Pll_Conf, 448
fatalfunc_t
algorithms_libs_rtklib, 277
fcb
nav_t, 762
fcbd_t, 480
bias, 481
std, 481
te, 481
ts, 481
fcorr
sbssatp_t, 941
FE_WGS84
algorithms_libs_rtklib, 281
fec.h, 1550, 1551
V27POLYA, 1551
V27POLYB, 1551
fft_base_kernels.h, 1174, 1175
fft_internal.h, 1178
kernel_dir_t, 1178
fft_size
Acq_Conf_Fpga, 338
fifo_reader.h, 1464
fifo_signal_source.h, 1407
FifoReader, 481
sptr, 482
start, 482

FifoSignalSource, 482
connect, 483
disconnect, 483
get_left_block, 483
get_right_block, 483
item_size, 483
file
ftp_t, 520
file_configuration.h, 1704, 1705
file_id
SensorDataConfiguration, 946
file_offset
SensorDataFileConfiguration, 947
file_signal_source.h, 1408, 1409
file_source_base.h, 1410
file_t, 484
fp, 484
fp_tag, 484
fp_tag_tmp, 484
fp_tmp, 484
fpos, 484
lock, 484
mode, 484
offset, 485
openpath, 485
path, 485
repmode, 485
speed, 485
start, 485
swapintv, 485
tick, 485
tick_f, 485
time, 485
timetag, 485
wtime, 485
file_timestamp_signal_source.h, 1413
FileConfiguration, 486
is_present, 486
property, 486–488
set_property, 488
filename
FileSourceBase, 493
GnssSdrLogSink, 644
GSS6450FileSignalSource, 695
MultichannelFileSignalSource, 757
SensorDataFileConfiguration, 947
FILEPATHSEP
algorithms_libs_rtklib, 281
files
rtksvr_t, 923
FileSignalSource, 488
FileSourceBase, 490
computeSamplesInFile, 493
connect, 493
disconnect, 493
filename, 493
FileSourceBase, 493
get_left_block, 493
get_right_block, 494
init, 494
item_size, 494
item_type, 494
itemTypeToSize, 494
packetsPerSample, 494
repeat, 494
samples, 494
samplesToSkip, 494
sampling_frequency, 494
source, 494
source_item_size, 495
FileTimestampSignalSource, 495
create_file_source_hook, 497
pre_connect_hook, 497
pre_disconnect_hook, 497
source, 497
find
Galileo_Ephemeris_Store, 539
find_uio_dev_file_name
core_libs, 88
findUtmZone
algorithms_libs, 241
fingerprint_sha256
Osnma_Public_Key_Material, 804
FIR_BUF_SIZE
signal_source_libs, 310
fir_filter.h, 1108, 1109
FirFilter, 497
∼FirFilter, 498
connect, 499
disconnect, 499
FirFilter, 498
get_left_block, 499
get_right_block, 499
implementation, 499
item_size, 499
role, 499
FIRST_RS_BIT
core_system_parameters, 173
FIRST_RS_BIT_AFTER_IODNAV
core_system_parameters, 173
fit
eph_t, 473
fit_interval_flag
Beidou_Dnav_Ephemeris, 386
Gps_Ephemeris, 666
five_bit_look_up
ion_gsms_stream_encodings.h, 1511
fix
ssat_t, 981
fixcnt
ambc_t, 354
flag
eph_t, 473
Ntrip_Rtk_Signal, 784
raw_t, 875
solstat_t, 976

stec_t, 987
flag_all_ephemeris
Galileo_Ephemeris, 537
Flag_carrier_phase_continuous
Gnss_Synchro, 635
Flag_cycle_slip
Gnss_Synchro, 635
flag_GGTO
Galileo_Utc_Model, 571
Flag_half_cycle_slip
Gnss_Synchro, 635
flag_nmea_tty_port
Pvt_Conf, 856
Flag_PLL_180_deg_phase_locked
Gnss_Synchro, 635
flag_rtcm_server
Pvt_Conf, 857
flag_rtcm_tty_port
Pvt_Conf, 857
flag_utc_model
Galileo_Utc_Model, 571
Flag_valid_acquisition
Gnss_Synchro, 635
Flag_valid_pseudorange
Gnss_Synchro, 635
Flag_valid_symbol_output
Gnss_Synchro, 635
Flag_valid_word
Gnss_Synchro, 636
flags
ambc_t, 354
Signal_Enabled_Flags, 961
flexiband_signal_source.h, 1414
FlexibandSignalSource, 499
connect, 500
disconnect, 500
get_left_block, 501
get_right_block, 501
item_size, 501
fll_bw_hz
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 455
fll_filter_order
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 455
fll_four_quadrant_atan
tracking_libs, 321
floor_div
galileo_tow_utils.h, 1544
gnss_tow, 325
floor_to_second_ms
galileo_tow_utils.h, 1544
gnss_tow, 325
flowgraph
ControlThread, 428
fmcomms2_signal_source.h, 1416
Fmcomms2SignalSource, 501
connect, 502
disconnect, 502
get_left_block, 502
get_right_block, 502
item_size, 502
fmcomms5_signal_source_fpga.h, 1418
Fmcomms5SignalSourceFPGA, 503
connect, 504
disconnect, 504
get_left_block, 504
get_right_block, 504
item_size, 504
fnav
Galileo_Ephemeris_Store, 539
FNAV_A0_4_LSB
core_system_parameters, 173
FNAV_A0G_4_LSB
core_system_parameters, 173
FNAV_A12_2_LSB
core_system_parameters, 173
FNAV_A1_4_LSB
core_system_parameters, 173
FNAV_A1G_4_LSB
core_system_parameters, 173
FNAV_AF0_1_LSB
core_system_parameters, 174
FNAV_AF0_5_LSB
core_system_parameters, 174
FNAV_AF1_1_LSB
core_system_parameters, 174
FNAV_AF1_5_LSB
core_system_parameters, 174
FNAV_AF2_1_LSB
core_system_parameters, 174
FNAV_AI0_1_LSB
core_system_parameters, 174
FNAV_AI1_1_LSB
core_system_parameters, 174
FNAV_AI2_1_LSB
core_system_parameters, 174
FNAV_BGD_1_LSB
core_system_parameters, 174
FNAV_CIC_4_LSB
core_system_parameters, 174
FNAV_CIS_4_LSB
core_system_parameters, 174
FNAV_CRC_3_LSB
core_system_parameters, 174
FNAV_CRS_3_LSB
core_system_parameters, 175
FNAV_CUC_3_LSB
core_system_parameters, 175
fnav_current_symbol_delay_ms
galileo_tow_utils.h, 1544
FNAV_CUS_3_LSB
core_system_parameters, 175
FNAV_DELTAA12_5_LSB
core_system_parameters, 175
FNAV_DELTAI_5_LSB

core_system_parameters, 175
FNAV_DELTAN_3_LSB
core_system_parameters, 175
FNAV_E_2_LSB
core_system_parameters, 175
FNAV_E_5_LSB
core_system_parameters, 175
FNAV_I0_3_LSB
core_system_parameters, 175
FNAV_IDOT_2_LSB
core_system_parameters, 175
FNAV_M0_2_LSB
core_system_parameters, 175
FNAV_M0_5_LSB
core_system_parameters, 175
FNAV_OMEGA0_2_LSB
core_system_parameters, 176
FNAV_OMEGA0_5_LSB
core_system_parameters, 176
FNAV_OMEGADOT_2_LSB
core_system_parameters, 176
FNAV_OMEGADOT_5_LSB
core_system_parameters, 176
FNAV_T0A_5_LSB
core_system_parameters, 176
FNAV_T0C_1_LSB
core_system_parameters, 176
FNAV_T0E_3_LSB
core_system_parameters, 176
FNAV_T0G_4_LSB
core_system_parameters, 176
FNAV_T0T_4_LSB
core_system_parameters, 176
FNAV_W_3_LSB
core_system_parameters, 176
FNAV_W_5_LSB
core_system_parameters, 176
folding_factor
Acq_Conf, 334
format
cl_fft_plan, 412
opt_t, 791
rtksvr_t, 923
four_bit_cpx_file_signal_source.h, 1420
four_bit_look_up
ion_gsms_stream_encodings.h, 1511
FourBitCpxFileSignalSource, 504
create_file_source_hook, 506
itemTypeToSize, 506
packetsPerSample, 506
pre_connect_hook, 506
pre_disconnect_hook, 507
source, 507
FP
ion_gsms_stream_encodings.h, 1511
fp
file_t, 484
fp_tag
file_t, 484
fp_tag_tmp
file_t, 484
fp_tmp
file_t, 484
Fpga_Acquisition, 507
~Fpga_Acquisition, 508
close_device, 508
configure_acquisition, 508
Fpga_Acquisition, 508
init, 508
open_device, 508
read_acquisition_results, 508
reset_acquisition, 509
run_acquisition, 509
set_block_exp, 509
set_doppler_sweep, 509
set_local_code, 509
stop_acquisition, 509
write_local_code, 509
fpga_acquisition.h, 1080
Fpga_buffer_monitor, 509
~Fpga_buffer_monitor, 510
check_buffer_overflow_and_monitor_buffer_status, 510
Fpga_buffer_monitor, 510
fpga_buffer_monitor.h, 1492, 1493
Fpga_DMA, 510
~Fpga_DMA, 511
DMA_close, 511
DMA_open, 511
DMA_write, 511
Fpga_DMA, 511
get_buffer_address, 511
get_buffer_size, 511
fpga_dma-proxy.h, 1494, 1495
buffer, 1495
length, 1495
proxy_status, 1495
Fpga_dynamic_bit_selection, 512
~Fpga_dynamic_bit_selection, 512
bit_selection, 512
Fpga_dynamic_bit_selection, 512
fpga_dynamic_bit_selection.h, 1496, 1497
fpga_integration_period
Dll_Pll_Conf_Fpga, 455
fpga_multicorrelator.h, 1615, 1616
Fpga_Multicorrelator_8sc, 512
~Fpga_Multicorrelator_8sc, 514
Carrier_wipeoff_multicorrelator_resampler, 514
disable_secondary_codes, 514
enable_secondary_codes, 514
Fpga_Multicorrelator_8sc, 514
free, 514
initialize_secondary_code, 514
lock_channel, 514
open_channel, 514
read_sample_counter, 515

set_initial_sample, 515
set_local_code_and_taps, 515
set_output_vectors, 515
set_secondary_code_lengths, 515
unlock_channel, 515
update_local_code, 515
update_prn_code_length, 515
Fpga_spidev, 516
~Fpga_spidev, 516
Fpga_spidev, 516
read_reg32, 516
SPI_close, 516
SPI_open, 516
write_reg32, 517
fpga_spidev.h, 1498
Fpga_Switch, 517
~Fpga_Switch, 517
Fpga_Switch, 517
set_switch_position, 517
fpga_switch.h, 1499, 1500
fpos
file_t, 484
frame
Sbas_Raw_Message, 934
free
Fpga_Multicorrelator_8sc, 514
freq
half_cyc_tag, 696
FREQ1
core_system_parameters, 176
FREQ1_BDS
core_system_parameters, 177
FREQ1_GLO
core_system_parameters, 177
FREQ2
core_system_parameters, 177
FREQ2_BDS
core_system_parameters, 177
FREQ2_GLO
core_system_parameters, 177
FREQ3_BDS
core_system_parameters, 177
FREQ3_GLO
core_system_parameters, 177
FREQ5
core_system_parameters, 177
FREQ6
core_system_parameters, 177
FREQ7
core_system_parameters, 177
FREQ8
core_system_parameters, 178
FREQ9
core_system_parameters, 178
freq_xlating_fir_filter.h, 1111
freqn
raw_t, 875
freqopt
prcopt_t, 847
FreqXlatingFirFilter, 518
connect, 518
disconnect, 518
get_left_block, 518
get_right_block, 519
implementation, 519
item_size, 519
role, 519
fresh
Ntrip_Rtcm_Snapshot, 782
FRIEND_TEST
osnma_msg_receiver.h, 1680
from_startup_load
Gnss_Almanac, 605
front
algorithms_libs, 241
frq
geph_t, 578
solstat_t, 976
fs
Gnss_Synchro, 636
fs_hz
stream_cfg, 988
fs_in
Acq_Conf, 334
Acq_Conf_Fpga, 338
Dll_Pll_Conf, 448
Dll_Pll_Conf_Fpga, 455
Kf_Conf, 727
ftime
rtksvr_t, 923
FTP_CMD
algorithms_libs_rtklib, 271
rtklib_stream.h, 1266
ftp_t, 519
addr, 519
error, 519
file, 520
local, 520
passwd, 520
proto, 520
state, 520
thread, 520
tnext, 520
topts, 520
user, 520
FTP_TIMEOUT
algorithms_libs_rtklib, 281
rtklib_stream.h, 1266
full_grid_search
Acq_Conf, 334

gal_almanac_map
Gnss_Sdr_Supl_Client, 627
gal_ephemeris_map
Gnss_Sdr_Supl_Client, 627
gal_iono
Gnss_Sdr_Supl_Client, 627

gal_utc
Gnss_Sdr_Supl_Client, 627
galhas_status
Monitor_Pvt, 750
Galileo_Almanac, 520
E1B_HS, 522
E5a_HS, 522
E5b_HS, 522
Galileo_Almanac, 522
IODa, 522
serialize, 522
galileo_almanac.h, 1781
Galileo_Almanac_Helper, 523
af0_10, 524
af0_8, 524
af0_9, 524
af1_10, 524
af1_8, 524
af1_9, 524
DELTA_A_7, 524
DELTA_A_8, 524
DELTA_A_9, 525
delta_i_7, 525
delta_i_8, 525
delta_i_9, 525
E1B_HS_10, 525
E1B_HS_8, 525
E1B_HS_9, 525
E5a_HS_10, 525
E5a_HS_8, 525
E5a_HS_9, 525
E5b_HS_10, 525
E5b_HS_8, 525
E5b_HS_9, 526
e_7, 526
e_8, 526
e_9, 526
Galileo_Almanac_Helper, 524
IOD_a_10, 526
IOD_a_7, 526
IOD_a_8, 526
IOD_a_9, 526
M0_10, 526
M0_7, 526
M0_9, 526
Omega0_10, 526
Omega0_7, 527
Omega0_8, 527
omega_7, 527
omega_8, 527
omega_9, 527
Omega_dot_10, 527
Omega_dot_7, 527
Omega_dot_8, 527
SVID1_7, 527
SVID2_8, 527
SVID3_9, 527
t0a_7, 527
t0a_9, 528
WN_a_7, 528
WN_a_9, 528
galileo_almanac_helper.h, 1782
galileo_almanac_map
Rtklib_Solver, 919
Galileo_CNAV.h, 1784, 1786
GALILEO_CNAV_BITS_FOR_CRC
core_system_parameters, 178
GALILEO_CNAV_BYTES_FOR_CRC
core_system_parameters, 178
GALILEO_CNAV_CRC_LENGTH
core_system_parameters, 178
GALILEO_CNAV_HAS_PAGE_DATA_BITS
core_system_parameters, 178
GALILEO_CNAV_INFORMATION_VECTOR_LENGTH
core_system_parameters, 178
GALILEO_CNAV_INTERLEAVER_COLS
core_system_parameters, 178
GALILEO_CNAV_INTERLEAVER_ROWS
core_system_parameters, 178
GALILEO_CNAV_MAX_NUMBER_SYMBOLS_ENCODED_BLOCK
core_system_parameters, 178
Galileo_Cnav_Message, 528
get_flag_CRC_test, 528
get_HAS_encoded_page, 528
have_new_HAS_page, 528
is_HAS_in_test_mode, 529
is_HAS_page_dummy, 529
set_time_stamp, 529
set_tow, 529
set_week, 529
galileo_cnav_message.h, 1788, 1789
GALILEO_CNAV_MESSAGE_BITS_PER_PAGE
core_system_parameters, 178
GALILEO_CNAV_MT1_HEADER_BITS
core_system_parameters, 179
GALILEO_CNAV_OCTETS_IN_SUBPAGE
core_system_parameters, 179
GALILEO_CNAV_PAGE_HEADER_BITS
core_system_parameters, 179
GALILEO_CNAV_PAGE_MS
core_system_parameters, 179
GALILEO_CNAV_PAGE_RESERVED_BITS
core_system_parameters, 179
GALILEO_CNAV_PREAMBLE
core_system_parameters, 179
GALILEO_CNAV_PREAMBLE_LENGTH_BITS
core_system_parameters, 179
GALILEO_CNAV_PREAMBLE_PERIOD_SYMBOLS
core_system_parameters, 179
GALILEO_CNAV_SYMBOLS_PER_PAGE
core_system_parameters, 179
GALILEO_CNAV_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 179
GALILEO_DATA_FRAME_BITS
core_system_parameters, 179
GALILEO_DATA_FRAME_BYTES

core_system_parameters, 179
GALILEO_DATA_JK_BITS
core_system_parameters, 180
Galileo_E1.h, 1790, 1791
GALILEO_E1_B_CODE_LENGTH_CHIPS
core_system_parameters, 180
GALILEO_E1_B_PRIMARY_CODE
core_system_parameters, 180
GALILEO_E1_B_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 180
GALILEO_E1_B_SAMPLES_PER_SYMBOL
core_system_parameters, 180
GALILEO_E1_B_SYMBOL_RATE_BPS
core_system_parameters, 180
GALILEO_E1_C_PRIMARY_CODE
core_system_parameters, 180
GALILEO_E1_C_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 180
GALILEO_E1_C_SECONDARY_CODE
core_system_parameters, 180
GALILEO_E1_C_SECONDARY_CODE_LENGTH
core_system_parameters, 180
GALILEO_E1_CODE_CHIP_RATE_CPS
core_system_parameters, 180
galileo_e1_code_gen_complex_sampled
algorithms_libs, 242
galileo_e1_code_gen_float_sampled
algorithms_libs, 242
galileo_e1_code_gen_sinboc11_float
algorithms_libs, 243
GALILEO_E1_CODE_PERIOD_MS
core_system_parameters, 181
GALILEO_E1_CODE_PERIOD_S
core_system_parameters, 181
GALILEO_E1_FREQ_HZ
core_system_parameters, 181
GALILEO_E1_HISTORY_DEEP
core_system_parameters, 181
GALILEO_E1_NUMBER_OF_CODES
core_system_parameters, 181
GALILEO_E1_OPT_ACQ_FS_SPS
core_system_parameters, 181
galileo_e1_signal_replica.h, 1138, 1139
GALILEO_E1_SUB_CARRIER_A_RATE_HZ
core_system_parameters, 181
GALILEO_E1_SUB_CARRIER_B_RATE_HZ
core_system_parameters, 181
galileo_e1_tcp_connector_tracking.h, 1566, 1567
Galileo_E1_Tcp_Connector_Tracking_cc, 529
galileo_e1_tcp_connector_tracking_cc.h, 1583, 1584
galileo_e1_tcp_connector_tracking_cc_sptr
tracking_gr_blocks, 316
galileo_e5_a_code_gen_complex_primary
algorithms_libs, 243
galileo_e5_a_code_gen_complex_sampled
algorithms_libs, 243
galileo_e5_b_code_gen_complex_primary
algorithms_libs, 243
galileo_e5_b_code_gen_complex_sampled
algorithms_libs, 243
galileo_e5_signal_replica.h, 1140, 1141
Galileo_E5a.h, 1808, 1809
GALILEO_E5A_CODE_CHIP_RATE_CPS
core_system_parameters, 181
GALILEO_E5A_CODE_LENGTH_CHIPS
core_system_parameters, 181
GALILEO_E5A_CODE_PERIOD_MS
core_system_parameters, 182
GALILEO_E5A_CODE_PERIOD_S
core_system_parameters, 182
GALILEO_E5A_CRC_ERROR_LIMIT
core_system_parameters, 182
GALILEO_E5A_FREQ_HZ
core_system_parameters, 182
GALILEO_E5A_HISTORY_DEEP
core_system_parameters, 182
GALILEO_E5A_I_PRIMARY_CODE
core_system_parameters, 182
GALILEO_E5A_I_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 182
GALILEO_E5A_I_SECONDARY_CODE
core_system_parameters, 182
GALILEO_E5A_I_SECONDARY_CODE_LENGTH
core_system_parameters, 182
GALILEO_E5A_I_TIERED_CODE_PERIOD_S
core_system_parameters, 182
galileo_e5a_noncoherent_iq_acquisition_caf_cc.h,
1040
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 530
~galileo_e5a_noncoherentIQ_acquisition_caf_cc,
531
general_work, 531
mag, 531
set_active, 531
set_channel, 531
set_channel_fsm, 532
set_gnss_synchro, 532
set_local_code, 532
galileo_e5a_noncoherentIQ_acquisition_caf_cc_sptr
acquisition_gr_blocks, 81
GALILEO_E5A_NUMBER_OF_CODES
core_system_parameters, 182
GALILEO_E5A_OPT_ACQ_FS_SPS
core_system_parameters, 183
GALILEO_E5A_Q_PRIMARY_CODE
core_system_parameters, 183
GALILEO_E5A_Q_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 183
GALILEO_E5A_Q_SECONDARY_CODE
core_system_parameters, 183
GALILEO_E5A_Q_SECONDARY_CODE_LENGTH
core_system_parameters, 183
GALILEO_E5A_Q_SECONDARY_CODE_STR_LENGTH
core_system_parameters, 183
GALILEO_E5A_Q_TIERED_CODE_PERIOD_S
core_system_parameters, 183

GALILEO_E5A_SYMBOL_RATE_BPS
core_system_parameters, 183
Galileo_E5b.h, 1852, 1853
GALILEO_E5B_CODE_CHIP_RATE_CPS
core_system_parameters, 183
GALILEO_E5B_CODE_LENGTH_CHIPS
core_system_parameters, 183
GALILEO_E5B_CODE_PERIOD_MS
core_system_parameters, 183
GALILEO_E5B_CODE_PERIOD_S
core_system_parameters, 184
GALILEO_E5B_FREQ_HZ
core_system_parameters, 184
GALILEO_E5B_HISTORY_DEEP
core_system_parameters, 184
GALILEO_E5B_I_PRIMARY_CODE
core_system_parameters, 184
GALILEO_E5B_I_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 184
GALILEO_E5B_I_SECONDARY_CODE
core_system_parameters, 184
GALILEO_E5B_I_SECONDARY_CODE_LENGTH
core_system_parameters, 184
GALILEO_E5B_I_TIERED_CODE_PERIOD_S
core_system_parameters, 184
GALILEO_E5B_NUMBER_OF_CODES
core_system_parameters, 184
GALILEO_E5B_OPT_ACQ_FS_SPS
core_system_parameters, 184
GALILEO_E5B_Q_PRIMARY_CODE
core_system_parameters, 184
GALILEO_E5B_Q_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 185
GALILEO_E5B_Q_SECONDARY_CODE
core_system_parameters, 185
GALILEO_E5B_Q_SECONDARY_CODE_LENGTH
core_system_parameters, 185
GALILEO_E5B_Q_SECONDARY_CODE_STR_LENGTH
core_system_parameters, 185
GALILEO_E5B_Q_TIERED_CODE_PERIOD_S
core_system_parameters, 185
GALILEO_E5B_SYMBOL_RATE_BPS
core_system_parameters, 185
Galileo_E6.h, 1895, 1896
GALILEO_E6_B_CODE_CHIP_RATE_CPS
core_system_parameters, 185
galileo_e6_b_code_gen_complex_primary
algorithms_libs, 243
galileo_e6_b_code_gen_complex_sampled
algorithms_libs, 244
galileo_e6_b_code_gen_float_primary
algorithms_libs, 244
GALILEO_E6_B_CODE_LENGTH_CHIPS
core_system_parameters, 185
GALILEO_E6_B_PRIMARY_CODE
core_system_parameters, 185
GALILEO_E6_B_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 185
GALILEO_E6_C_CODE_CHIP_RATE_CPS
core_system_parameters, 185
galileo_e6_c_code_gen_complex_primary
algorithms_libs, 244
galileo_e6_c_code_gen_complex_sampled
algorithms_libs, 244
galileo_e6_c_code_gen_float_primary
algorithms_libs, 244
GALILEO_E6_C_CODE_LENGTH_CHIPS
core_system_parameters, 186
GALILEO_E6_C_PRIMARY_CODE
core_system_parameters, 186
GALILEO_E6_C_PRIMARY_CODE_STR_LENGTH
core_system_parameters, 186
GALILEO_E6_C_SECONDARY_CODE
core_system_parameters, 186
galileo_e6_c_secondary_code
algorithms_libs, 244
galileo_e6_c_secondary_code_gen_complex
algorithms_libs, 245
galileo_e6_c_secondary_code_gen_float
algorithms_libs, 245
GALILEO_E6_C_SECONDARY_CODE_LENGTH_CHIPS
core_system_parameters, 186
GALILEO_E6_C_SECONDARY_CODE_STR_LENGTH
core_system_parameters, 186
GALILEO_E6_CODE_PERIOD_MS
core_system_parameters, 186
GALILEO_E6_CODE_PERIOD_S
core_system_parameters, 186
GALILEO_E6_FREQ_HZ
core_system_parameters, 186
galileo_e6_has_msg_receiver, 532
~galileo_e6_has_msg_receiver, 533
process_test_page, 533
galileo_e6_has_msg_receiver.h, 1651, 1652
galileo_e6_has_msg_receiver_sptr
core_libs, 87
GALILEO_E6_NUMBER_OF_CODES
core_system_parameters, 186
GALILEO_E6_OPT_ACQ_FS_SPS
core_system_parameters, 186
galileo_e6_signal_replica.h, 1142, 1143
Galileo_Ephemeris, 533
BGD_E1E5a, 536
BGD_E1E5b, 537
E1B_DVS, 537
E1B_HS, 537
E5a_DVS, 537
E5a_HS, 537
E5b_DVS, 537
E5b_HS, 537
flag_all_ephemeris, 537
Galileo_Ephemeris, 536
Galileo_System_Time, 536
IOD_ephemeris, 537
IOD_nav, 538
nav_message_source, 538

nav_message_type, 538
serialize, 536
SISA, 538
galileo_ephemeris.h, 1918, 1919
galileo_ephemeris_map
Rtklib_Solver, 919
Galileo_Ephemeris_Store, 538
by_source, 539
clear, 539
combined_view, 539
empty, 539
Ephemeris_Map, 539
find, 539
fnav, 539
inav, 539
insert, 539
galileo_ephemeris_store
Rtklib_Solver, 919
galileo_ephemeris_store.h, 1305, 1306
GALILEO_F
core_system_parameters, 187
Galileo_FNAV.h, 1921, 1924
GALILEO_FNAV_CODES_PER_PAGE
core_system_parameters, 187
GALILEO_FNAV_CODES_PER_PREAMBLE
core_system_parameters, 187
GALILEO_FNAV_CODES_PER_SYMBOL
core_system_parameters, 187
GALILEO_FNAV_DATA_FRAME_BITS
core_system_parameters, 187
GALILEO_FNAV_DATA_FRAME_BYTES
core_system_parameters, 187
GALILEO_FNAV_INTERLEAVER_COLS
core_system_parameters, 187
GALILEO_FNAV_INTERLEAVER_ROWS
core_system_parameters, 187
Galileo_Fnav_Message, 540
get_flag_CRC_test, 540
get_flag_TOW_set, 540
get_TOW1, 541
get_TOW2, 541
get_TOW3, 541
get_TOW4, 541
get_WN1, 541
get_WN2, 541
get_WN3, 541
is_TOW1_set, 541
is_TOW2_set, 541
is_TOW3_set, 541
is_TOW4_set, 541
set_flag_TOW_set, 541
set_TOW1_flag, 542
set_TOW2_flag, 542
set_TOW3_flag, 542
set_TOW4_flag, 542
galileo_fnav_message.h, 1926, 1927
GALILEO_FNAV_PAGE_TYPE_BITS
core_system_parameters, 187
GALILEO_FNAV_PREAMBLE
core_system_parameters, 187
GALILEO_FNAV_PREAMBLE_LENGTH_BITS
core_system_parameters, 187
GALILEO_FNAV_SECONDS_PER_PAGE
core_system_parameters, 187
GALILEO_FNAV_SYMBOLS_PER_PAGE
core_system_parameters, 188
GALILEO_GM
core_system_parameters, 188
Galileo_HAS_data, 542
cell_mask, 550
cell_mask_availability_flag, 550
code_bias, 550
delta_clock_correction, 550
delta_clock_correction_clock_subset, 550
delta_clock_multiplier, 550
delta_clock_multiplier_clock_subset, 550
delta_cross_track, 550
delta_in_track, 550
delta_radial, 551
get_clock_correction_mult_m, 545
get_clock_subset_correction_mult_m, 545
get_code_bias_m, 546
get_delta_clock_correction_m, 546
get_delta_clock_subset_correction_m, 546
get_delta_cross_track_m, 546
get_delta_in_track_m, 546, 547
get_delta_radial_m, 547
get_gnss_id, 547
get_gnss_iod, 547
get_nsat, 547
get_nsat_sub, 547
get_num_satellites, 548
get_num_subset_satellites, 548
get_phase_bias_cycle, 548
get_phase_discontinuity_indicator, 548
get_PRNs_in_mask, 548
get_PRNs_in_submask, 548
get_signals_in_mask, 548
get_systems_string, 549
get_systems_subset_string, 549
get_time_of_message_s, 549
get_validity_interval_s, 549
gnss_id_clock_subset, 551
gnss_id_mask, 551
gnss_iod, 551
has_status, 551
has_validity_interval_at_tow, 549
header, 551
is_clock_do_not_use, 549
is_clock_unavailable, 549
is_code_bias_unavailable, 549
is_orbit_along_cross_unavailable, 549
is_orbit_radial_unavailable, 549
is_phase_bias_unavailable, 550
message_id, 551
nav_message, 551

Nsys, 551
Nsys_sub, 551
phase_bias, 552
phase_discontinuity_indicator, 552
satellite_mask, 552
satellite_submask, 552
signal_mask, 552
tow, 552
validity_interval_index_clock_fullset_corrections, 552
validity_interval_index_clock_subset_corrections, 552
validity_interval_index_code_bias_corrections, 552
validity_interval_index_orbit_corrections, 552
validity_interval_index_phase_bias_corrections, 553
week, 553
galileo_has_data.h, 1930, 1931
GALILEO_HAS_HALF_WEEK_SECONDS
core_system_parameters, 188
GALILEO_HAS_INVALID_TOW
core_system_parameters, 188
GALILEO_HAS_INVALID_WEEK
core_system_parameters, 188
Galileo_HAS_page, 553
has_message_string, 554
has_status, 554
message_id, 554
message_page_id, 554
message_size, 554
message_type, 554
reserved, 554
time_stamp, 554
tow, 554
week, 554
galileo_has_page.h, 1933, 1934
GALILEO_HAS_SECONDS_PER_HOUR
core_system_parameters, 188
GALILEO_HAS_SECONDS_PER_WEEK
core_system_parameters, 188
Galileo_INAV.h, 1934, 1939
GALILEO_INAV_DUMMY_WORD_TYPE
core_system_parameters, 188
GALILEO_INAV_ENCODED_SSP1
core_system_parameters, 188
GALILEO_INAV_ENCODED_SSP2
core_system_parameters, 188
GALILEO_INAV_ENCODED_SSP3
core_system_parameters, 188
GALILEO_INAV_EVEN_PAGE_TYPE_BIT
core_system_parameters, 188
GALILEO_INAV_INTERLEAVER_COLS
core_system_parameters, 189
GALILEO_INAV_INTERLEAVER_ROWS
core_system_parameters, 189
Galileo_Inav_Message, 555
enable_reed_solomon, 556
get_A0G, 556
get_A1G, 556
get_flag_CRC_test, 556
get_flag_GGTO, 556
get_flag_TOW_set, 556
get_Galileo_week, 556
get_t0G, 556
get_TOW0, 556
get_TOW5, 556
get_TOW6, 556
get_WN0G, 557
init_PRN, 557
is_alert_page, 557
is_TOW0_set, 557
is_TOW5_set, 557
is_TOW6_set, 557
set_flag_TOW_set, 557
set_TOW0_flag, 557
set_TOW5_flag, 557
set_TOW6_flag, 557
galileo_inav_message.h, 1943, 1944
GALILEO_INAV_ODD_PAGE_TYPE_BIT
core_system_parameters, 189
GALILEO_INAV_PAGE_PART_MS
core_system_parameters, 189
GALILEO_INAV_PAGE_PART_SECONDS
core_system_parameters, 189
GALILEO_INAV_PAGE_PART_SYMBOLS
core_system_parameters, 189
GALILEO_INAV_PAGE_PART_WITH_PREABLE_SECONDS
core_system_parameters, 189
GALILEO_INAV_PAGE_SECONDS
core_system_parameters, 189
GALILEO_INAV_PAGE_SYMBOLS
core_system_parameters, 189
GALILEO_INAV_PLAIN_SSP1
core_system_parameters, 189
GALILEO_INAV_PLAIN_SSP2
core_system_parameters, 189
GALILEO_INAV_PLAIN_SSP3
core_system_parameters, 190
GALILEO_INAV_PREAMBLE
core_system_parameters, 190
GALILEO_INAV_PREAMBLE_LENGTH_BITS
core_system_parameters, 190
GALILEO_INAV_PREAMBLE_PERIOD_SYMBOLS
core_system_parameters, 190
Galileo_Iono, 558
ai0, 559
ai1, 559
ai2, 559
Galileo_Iono, 558
Region1_flag, 559
Region2_flag, 559
Region3_flag, 559
Region4_flag, 559
Region5_flag, 560
serialize, 559
tow, 560

valid, 560
WN, 560
galileo_iono
Rtklib_Solver, 919
galileo_iono.h, 1950, 1951
Galileo_ISM, 560
Galileo_ISM, 561
galileo_ism.h, 1952
GALILEO_ISM_CRC_DATA_BITS
core_system_parameters, 190
GALILEO_ISM_CRC_DATA_BYTES
core_system_parameters, 190
Galileo_Nav_Message_Source
core_system_parameters, 133
Galileo_Nav_Message_Type
core_system_parameters, 133
Galileo_OSNMA.h, 1955, 1956
GALILEO_PAGE_TYPE_BITS
core_system_parameters, 190
galileo_pcps_8ms_acquisition_cc, 561
~galileo_pcps_8ms_acquisition_cc, 562
general_work, 562
mag, 562
set_active, 562
set_channel, 563
set_channel_fsm, 563
set_gnss_synchro, 563
set_local_code, 563
galileo_pcps_8ms_acquisition_cc.h, 1043, 1044
galileo_pcps_8ms_acquisition_cc_sptr
acquisition_gr_blocks, 81
Galileo_Reduced_CED, 564
af0red, 565
af1red, 565
BGD_E1E5b, 565
CED_VALIDITY_SECONDS, 565
compute_eph, 565
DeltaAred, 565
Deltai0red, 565
E1B_DVS, 566
E1B_HS, 566
E5b_DVS, 566
E5b_HS, 566
exred, 566
eyred, 566
Galileo_Reduced_CED, 565
is_valid_at, 565
lambda0red, 566
Omega0red, 566
PRN, 566
reference_time, 565
TOTRedCED, 566
WN, 567
galileo_reduced_ced.h, 1958, 1959
galileo_reduced_ced_map
Rtklib_Solver, 919
galileo_satellites
sdr_gnss_packet_t, 942
GALILEO_SECONDS_PER_WEEK
osnma_data.h, 2044
Galileo_System_Time
Galileo_Ephemeris, 536
galileo_telemetry_decoder_gs, 567
general_work, 568
reset, 568
set_channel, 568
set_satellite, 568
galileo_telemetry_decoder_gs.h, 1527, 1528
galileo_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
GALILEO_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 190
galileo_tow_map, 568
~galileo_tow_map, 569
galileo_tow_map.h, 1654, 1655
GALILEO_TOW_MAP_INVALID_SAMPLE_COUNTER
core_libs, 88
GALILEO_TOW_MAP_INVALID_TOW_MS
core_libs, 88
GALILEO_TOW_MAP_INVALID_WEEK
core_libs, 88
galileo_tow_map_sptr
core_libs, 87
GALILEO_TOW_MAP_WEEK_MS
core_libs, 88
galileo_tow_utils.h, 1543, 1545
add_ms, 1544
floor_div, 1544
floor_to_second_ms, 1544
fnav_current_symbol_delay_ms, 1544
inav_current_symbol_delay_ms, 1544
sample_counter_delta, 1544
seconds_to_ms, 1544
week_after_delta, 1544
WEEK_MS, 1545
wrap_ms, 1544
Galileo_Utc_Model, 569
A0, 570
A1, 570
A_0G, 570
A_1G, 570
Delta_tLS, 571
Delta_tLSF, 571
DN, 571
flag_GGTO, 571
flag_utc_model, 571
Galileo_Utc_Model, 570
GST_to_UTC_time, 570
serialize, 570
t_0G, 571
tot, 571
truncated_week_diff, 570
WN_0G, 571
WN_LSF, 571
WNot, 571
galileo_utc_model

Rtklib_Solver, 919
galileo_utc_model.h, 1960
GalileoE1TcpConnectorTracking, 572
connect, 572
disconnect, 572
get_left_block, 573
get_right_block, 573
implementation, 573
item_size, 573
role, 573
set_channel, 573
set_gnss_synchro, 573
start_tracking, 573
stop_tracking, 573
GalileoTowMap
core_libs, 87
GalileoTowMapEntry, 574
GalileoTowMapEntry, 574
sample_counter, 574
tow_ms, 574
week, 574
GalileoTowMapMessage
core_libs, 87
gamn
geph_t, 579
GAP_RESION
algorithms_libs_rtklib, 281
gdop
Monitor_Pvt, 751
gen_signal_source.h, 1421, 1422
general_work
beidou_dnav_telemetry_decoder_gs, 395
dll_pll_veml_tracking_fpga, 459
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 531
galileo_pcps_8ms_acquisition_cc, 562
galileo_telemetry_decoder_gs, 568
glonass_gnav_telemetry_decoder_gs, 598
gps_l1_ca_telemetry_decoder_gs, 673
gps_l2c_telemetry_decoder_gs, 675
pcps_acquisition, 809
pcps_acquisition_fine_doppler_cc, 812
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 819
pcps_opencl_acquisition_cc, 822
pcps_quicksync_acquisition_cc, 824
pcps_tong_acquisition_cc, 827
sbas_l1_telemetry_decoder_gs, 933
generation
Ntrip_Rtcm_Snapshot, 783
GenSignalSource, 574
connect, 575
disconnect, 575
GenSignalSource, 575
get_left_block, 576
get_right_block, 576
getRfChannels, 576
implementation, 576
item_size, 576
role, 576
signal_generator, 576
Geo_to_ECEF
algorithms_libs, 245
geofunctions.h, 1144, 1145
Geohash, 576
decode, 577
encode, 577
geohash
Monitor_Pvt, 751
geohash.h, 1308
geoid
solopt_t, 974
geojson_output_enabled
Pvt_Conf, 857
geojson_output_path
Pvt_Conf, 857
GeoJSON_Printer, 577
geojson_printer.h, 1309
geojson_rate_ms
Pvt_Conf, 857
geph
nav_t, 762
geph_t, 578
acc, 578
age, 578
dtaun, 578
frq, 578
gamn, 579
iode, 579
pos, 579
sat, 579
sva, 579
svh, 579
taun, 579
toe, 579
tof, 579
vel, 579
Get
INIReader, 712
get
algorithms_libs, 245
Protobuf_Cleanup_Manager, 851
get_A0G
Galileo_Inav_Message, 556
get_A1G
Galileo_Inav_Message, 556
get_accumulated_tag_adkd
OSNMA_NavData, 801
get_alm_satellite_slot_number
Glonass_Gnav_Navigation_Message, 594
get_almanac
Beidou_Dnav_Navigation_Message, 392
Glonass_Gnav_Navigation_Message, 594
Gps_Navigation_Message, 677
get_almanac_config_code
Gps_Navigation_Message, 677

get_almanac_health
Gps_Navigation_Message, 678
get_beidou_cnav1_ephemeris
PvtInterface, 867
Rtklib_Pvt, 909
get_beidou_cnav2_ephemeris
PvtInterface, 867
Rtklib_Pvt, 909
get_beidou_dnav_almanac
Rtklib_Pvt, 909
get_beidou_dnav_almanac_map
rtklib_pvt_gs, 913
get_beidou_dnav_ephemeris
Rtklib_Pvt, 909
get_beidou_dnav_ephemeris_map
rtklib_pvt_gs, 913
get_block
Gnss_Satellite, 619
get_broadcast_signal_health
Rtklib_Solver, 917
get_buffer_address
Fpga_DMA, 511
get_buffer_size
Fpga_DMA, 511
get_clock_correction_mult_m
Galileo_HAS_data, 545
get_clock_drift_ppm
Pvt_Solution, 865
get_clock_subset_correction_mult_m
Galileo_HAS_data, 545
get_code_bias_m
Galileo_HAS_data, 546
get_code_nco
Tracking_2nd_DLL_filter, 1009
get_course_over_ground
Pvt_Solution, 865
get_current_status_map
channel_status_msg_receiver, 409
get_current_status_pvt
channel_status_msg_receiver, 409
get_current_tracking_map
channel_status_msg_receiver, 409
get_delta_clock_correction_m
Galileo_HAS_data, 546
get_delta_clock_subset_correction_m
Galileo_HAS_data, 546
get_delta_cross_track_m
Galileo_HAS_data, 546
get_delta_in_track_m
Galileo_HAS_data, 546, 547
get_delta_radial_m
Galileo_HAS_data, 547
get_differential_corrections
Beidou_Dnav_Navigation_Message, 392
get_eop
Gps_CNAV_Navigation_Message, 660
get_ephemeris
Beidou_Cnav2_Navigation_Message, 377
Beidou_Dnav_Navigation_Message, 392
Glonass_Gnav_Navigation_Message, 594
Gps_CNAV_Navigation_Message, 660
Gps_Navigation_Message, 678
get_epoch_count
HistogramBitSynchronizer, 703
get_first_accumulated_tag_gst
OSNMA_NavData, 801
get_flag_CRC_test
Beidou_Dnav_Navigation_Message, 392
Galileo_Cnav_Message, 528
Galileo_Fnav_Message, 540
Galileo_Inav_Message, 556
Glonass_Gnav_Navigation_Message, 595
get_flag_GGTO
Galileo_Inav_Message, 556
get_flag_iono_valid
Gps_Navigation_Message, 678
get_flag_new_SOW_available
Beidou_Dnav_Navigation_Message, 392
get_flag_TOW_new
Glonass_Gnav_Navigation_Message, 595
get_flag_TOW_set
Galileo_Fnav_Message, 540
Galileo_Inav_Message, 556
get_flag_update_slot_number
Glonass_Gnav_Navigation_Message, 595
get_flag_utc_model_valid
Gps_Navigation_Message, 678
get_frame_number
Glonass_Gnav_Navigation_Message, 595
get_galileo_almanac
Rtklib_Pvt, 910
get_galileo_almanac_map
rtklib_pvt_gs, 913
get_galileo_ephemeris
Rtklib_Pvt, 910
get_galileo_ephemeris_map
rtklib_pvt_gs, 913
get_Galileo_week
Galileo_Inav_Message, 556
get_gdop
Rtklib_Solver, 917
get_glonass_almanac
Rtklib_Pvt, 910
get_glonass_ephemeris
Rtklib_Pvt, 910
get_glonass_utc_model
Rtklib_Pvt, 910
get_gnss_id
Galileo_HAS_data, 547
get_gnss_iod
Galileo_HAS_data, 547
get_GPGGA
Nmea_Printer, 767
get_GPGSA
Nmea_Printer, 767
get_gps_almanac

Rtklib_Pvt, 910
get_gps_almanac_map
rtklib_pvt_gs, 913
get_gps_cnav_ephemeris
Rtklib_Pvt, 910
get_gps_cnav_ephemeris_map
rtklib_pvt_gs, 913
get_gps_ephemeris
Rtklib_Pvt, 910
get_GPS_week
Gps_Navigation_Message, 678
get_HAS_encoded_page
Galileo_Cnav_Message, 528
get_hdop
Rtklib_Solver, 917
get_height
Pvt_Solution, 865
get_histogram
HistogramBitSynchronizer, 703
get_IOD_nav
OSNMA_NavData, 801
get_iono
Beidou_Dnav_Navigation_Message, 392
Gps_CNAV_Navigation_Message, 660
Gps_Navigation_Message, 678
get_last_accumulated_tag_gst
OSNMA_NavData, 801
get_last_nav_bits
Beidou_Cnav1_Navigation_Message, 375
Beidou_Cnav2_Navigation_Message, 377
get_last_received_TOW
OSNMA_NavData, 801
get_last_received_WN
OSNMA_NavData, 801
get_latest_PVT
Rtklib_Pvt, 910
rtklib_pvt_gs, 913
get_latitude
Pvt_Solution, 865
get_left_block
Ad936xCustomSignalSource, 347
Adrv9361z7035SignalSourceFPGA, 348
ArraySignalConditioner, 357
BeamformerFilter, 366
BladerfSignalSource, 401
ByteToShort, 402
Channel, 405
ChannelInterface, 410
CshortToGrComplex, 433
CustomUDPSignalSource, 438
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
DMASignalSourceFPGA, 465
FifoSignalSource, 483
FileSourceBase, 493
FirFilter, 499
FlexibandSignalSource, 501
Fmcomms2SignalSource, 502
Fmcomms5SignalSourceFPGA, 504
FreqXlatingFirFilter, 518
GalileoE1TcpConnectorTracking, 573
GenSignalSource, 576
GnMaxSignalSource, 602
GNSSBlockInterface, 640
GpsL1CaDllPllTrackingGPU, 682
GpsL1CaGaussianTracking, 684
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
GSS6450FileSignalSource, 695
HybridObservables, 706
IbyteToCbyte, 707
IbyteToComplex, 709
IbyteToCshort, 710
IONGSMSSignalSource, 720
IshortToComplex, 721
IshortToCshort, 723
LabsatSignalSource, 732
LimesdrSignalSource, 737
MAX2771EVKITSignalSourceFPGA, 746
MmseResamplerConditioner, 747
MultichannelFileSignalSource, 757
NotchFilter, 769
NotchFilterLite, 770
NTLabFileSignalSource, 777
OsmosdrSignalSource, 793
Pass_Through, 806
PcpsAcquisitionAdapter, 829
PcpsAcquisitionAdapterCustom, 832
PcpsAcquisitionAdapterFpga, 835
PlutosdrSignalSource, 840
PocketSdrSignalSource, 842
PulseBlankingFilter, 853
RawArraySignalSource, 879
Rtklib_Pvt, 910
RtlTcpSignalSource, 928
SignalConditioner, 965
SignalGenerator, 966
SignalSourceBase, 969
TelemetryDecoderAdapter, 1000
UhdSignalSource, 1021
get_left_block_acq
Channel, 405
ChannelInterface, 410
get_left_block_trk
Channel, 405
ChannelInterface, 411
get_LLH
TcpCmdInterface, 996
get_longitude
Pvt_Solution, 865
get_map_copy
Concurrent_Map< Data >, 420
get_merkle_root
Gnss_Crypto, 609
get_navfilename

Rinex_Printer, 882
get_nsat
Galileo_HAS_data, 547
get_nsat_sub
Galileo_HAS_data, 547
get_num_devices
core_libs, 88
get_num_satellites
Galileo_HAS_data, 548
get_num_subset_satellites
Galileo_HAS_data, 548
get_num_valid_observations
Pvt_Solution, 865
get_obsfilename
Rinex_Printer, 882
get_pdop
Rtklib_Solver, 917
get_phase_bias_cycle
Galileo_HAS_data, 548
get_phase_discontinuity_indicator
Galileo_HAS_data, 548
get_PRN
Gnss_Satellite, 619
get_prn_d
OSNMA_NavData, 801
get_PRNs_in_mask
Galileo_HAS_data, 548
get_PRNs_in_submask
Galileo_HAS_data, 548
get_public_key_compressed
Gnss_Crypto, 609
get_public_key_type
Gnss_Crypto, 609
get_pvt
GNSSFlowgraph, 642
get_ref_gps_week
Pvt_Solution, 865
get_rf_link
Gnss_Satellite, 619
get_right_block
Ad936xCustomSignalSource, 347
Adrv9361z7035SignalSourceFPGA, 348
ArraySignalConditioner, 357
BeamformerFilter, 366
BladerfSignalSource, 401
ByteToShort, 402
Channel, 405
ChannelInterface, 411
CshortToGrComplex, 433
CustomUDPSignalSource, 438
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
DMASignalSourceFPGA, 465
Evk1029SignalSource, 477
FifoSignalSource, 483
FileSourceBase, 494
FirFilter, 499
FlexibandSignalSource, 501
Fmcomms2SignalSource, 502
Fmcomms5SignalSourceFPGA, 504
FreqXlatingFirFilter, 519
GalileoE1TcpConnectorTracking, 573
GenSignalSource, 576
GnMaxSignalSource, 602
GNSSBlockInterface, 640
GpsL1CaDllPllTrackingGPU, 682
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
GSS6450FileSignalSource, 695
HybridObservables, 706
IbyteToCbyte, 708
IbyteToComplex, 709
IbyteToCshort, 710
IONGSMSSignalSource, 720
IshortToComplex, 721
IshortToCshort, 723
LabsatSignalSource, 732
LimesdrSignalSource, 737
MAX2771EVKITSignalSourceFPGA, 746
MmseResamplerConditioner, 747
MultichannelFileSignalSource, 757
NotchFilter, 769
NotchFilterLite, 771
NTLabFileSignalSource, 777
OsmosdrSignalSource, 793
Pass_Through, 806
PcpsAcquisitionAdapter, 829
PcpsAcquisitionAdapterCustom, 832
PcpsAcquisitionAdapterFpga, 835
PlutosdrSignalSource, 840
PocketSdrSignalSource, 842
PulseBlankingFilter, 854
RawArraySignalSource, 879
Rtklib_Pvt, 911
RtlTcpSignalSource, 928
SignalConditioner, 965
SignalGenerator, 966
TelemetryDecoderAdapter, 1000
UhdSignalSource, 1021
ZmqSignalSource, 1032
get_right_block_acq
Channel, 405
ChannelInterface, 411
get_right_block_trk
Channel, 405
ChannelInterface, 411
get_role_name
block_factory, 323
get_satellite
Gnss_Signal, 630
get_satellite_PRN
Gps_Navigation_Message, 678
get_signal
Channel, 405

get_signal_str
Gnss_Signal, 630
get_signals_in_mask
Galileo_HAS_data, 548
get_SOW
Beidou_Dnav_Navigation_Message, 392
get_speed_over_ground
Pvt_Solution, 865
get_system
Gnss_Satellite, 620
Gps_Almanac, 649
Gps_Ephemeris, 665
get_system_short
Gnss_Satellite, 620
get_systems_string
Galileo_HAS_data, 549
get_systems_subset_string
Galileo_HAS_data, 549
get_t0G
Galileo_Inav_Message, 556
get_time_of_message_s
Galileo_HAS_data, 549
get_time_offset_s
Pvt_Solution, 865
get_total_events
HistogramBitSynchronizer, 703
get_TOW
Gps_Navigation_Message, 678
get_TOW0
Galileo_Inav_Message, 556
get_TOW1
Galileo_Fnav_Message, 541
get_TOW2
Galileo_Fnav_Message, 541
get_TOW3
Galileo_Fnav_Message, 541
get_TOW4
Galileo_Fnav_Message, 541
get_TOW5
Galileo_Inav_Message, 556
get_TOW6
Galileo_Inav_Message, 556
get_tow_sf0
OSNMA_NavData, 802
get_tuner_gain_count
Rtl_Tcp_Dongle_Info, 926
get_tuner_type
Rtl_Tcp_Dongle_Info, 926
get_utc_model
Beidou_Dnav_Navigation_Message, 392
Glonass_Gnav_Navigation_Message, 595
Gps_CNAV_Navigation_Message, 660
Gps_Navigation_Message, 678
get_utc_time
TcpCmdInterface, 996
get_validity_interval_s
Galileo_HAS_data, 549
get_vdop
Rtklib_Solver, 917
get_verified_bits
OSNMA_NavData, 802
get_verified_status
OSNMA_NavData, 802
get_WN0G
Galileo_Inav_Message, 557
get_WN1
Galileo_Fnav_Message, 541
get_WN2
Galileo_Fnav_Message, 541
get_WN3
Galileo_Fnav_Message, 541
get_wn_sf0
OSNMA_NavData, 802
GetBlock
block_factory, 324
GetDwellResidualSamples
Acq_Conf, 332
GetInteger
INIReader, 712
getRfChannels
GenSignalSource, 576
GSS6450FileSignalSource, 695
IONGSMSSignalSource, 720
LabsatSignalSource, 732
SignalSourceBase, 969
GetSamplesPerDwell
Acq_Conf, 332
GetYIntercept
tracking_libs, 321
GetYInterceptAbs
tracking_libs, 321
gf
ssat_t, 981
gf2
ssat_t, 981
gga_period_ms
Ntrip_Rtcm_Client_Config, 779
give
sbsigp_t, 937
GIVEI
Beidou_Dnav_Iono_Grid_Point, 390
glo_cpbias
nav_t, 762
glo_fcn
nav_t, 763
glo_gnav_utc
Gnss_Sdr_Supl_Client, 627
gloicb
exterr_t, 480
glomodear
prcopt_t, 847
glonass_almanac_epoch
algorithms_libs_rtklib, 277
GLONASS_C20
core_system_parameters, 190
GLONASS_CA_NBR_SATS

core_system_parameters, 190
GLONASS_EARTH_INCLINATION
core_system_parameters, 190
GLONASS_EARTH_RADIUS
core_system_parameters, 190
GLONASS_F_M_A
core_system_parameters, 191
GLONASS_FLATTENING
core_system_parameters, 191
GLONASS_GM
core_system_parameters, 191
Glonass_Gnav_Almanac, 580
d_C_n, 581
d_Delta_i_n_A, 581
d_Delta_T_n_A, 581
d_Delta_T_n_A_dot, 582
d_epsilon_n_A, 582
d_H_n_A, 582
d_KP, 582
d_l_n, 582
d_lambda_n_A, 582
d_M_n_A, 582
d_N_4, 582
d_N_A, 583
d_n_A, 583
d_omega_n_A, 583
d_t_lambda_n_A, 583
d_tau_n_A, 583
Glonass_Gnav_Almanac, 581
i_satellite_freq_channel, 583
i_satellite_slot_number, 583
PRN, 583
satellite_position, 581
serialize, 581
glonass_gnav_almanac
Rtklib_Solver, 919
glonass_gnav_almanac.h, 1961, 1962
glonass_gnav_almanac_map
Rtklib_Solver, 919
GLONASS_GNAV_CRC_I_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_J_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_K_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_L_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_M_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_N_INDEX
core_system_parameters, 191
GLONASS_GNAV_CRC_P_INDEX
core_system_parameters, 192
GLONASS_GNAV_CRC_Q_INDEX
core_system_parameters, 192
GLONASS_GNAV_DATA_SYMBOLS
core_system_parameters, 192
GLONASS_GNAV_ECC_LOCATOR
core_system_parameters, 192
Glonass_Gnav_Ephemeris, 584
compute_GLONASS_time, 586
d_AXn, 588
d_AYn, 588
d_AZn, 588
d_B_n, 588
d_Delta_tau_n, 588
d_dtr, 589
d_E_n, 589
d_F_T, 589
d_gamma_n, 589
d_iode, 589
d_kp_leap_correction_s, 589
d_l3rd_n, 589
d_l5th_n, 589
d_M, 590
d_m, 590
d_n, 590
d_N_T, 590
d_P, 590
d_P_1, 590
d_P_2, 590
d_P_3, 590
d_P_4, 591
d_satClkDrift, 591
d_t_b, 591
d_t_k, 591
d_tau_c, 591
d_tau_n, 591
d_tod, 591
d_TOW, 591
d_VXn, 592
d_VYn, 592
d_VZn, 592
d_WN, 592
d_Xn, 592
d_Yn, 592
d_yr, 592
d_Zn, 592
Glonass_Gnav_Ephemeris, 586
glot_to_gpst, 586
glot_to_utc, 587
i_satellite_freq_channel, 592
i_satellite_slot_number, 593
PRN, 593
serialize, 587
sv_clock_drift, 588
glonass_gnav_ephemeris.h, 1964
glonass_gnav_ephemeris_map
Gnss_Sdr_Supl_Client, 627
Rtklib_Solver, 919
GLONASS_GNAV_HAMMING_CODE_BITS
core_system_parameters, 192
glonass_gnav_is_healthy
algorithms_libs_rtklib, 278
Glonass_Gnav_Navigation_Message, 593
CRC_test, 594

get_alm_satellite_slot_number, 594
get_almanac, 594
get_ephemeris, 594
get_flag_CRC_test, 595
get_flag_TOW_new, 595
get_flag_update_slot_number, 595
get_frame_number, 595
get_utc_model, 595
Glonass_Gnav_Navigation_Message, 594
have_new_almanac, 595
have_new_ephemeris, 595
have_new_utc_model, 595
is_flag_TOW_set, 595
set_flag_ephemeris_str_1, 595
set_flag_ephemeris_str_2, 596
set_flag_ephemeris_str_3, 596
set_flag_ephemeris_str_4, 596
set_flag_TOW_new, 596
set_flag_TOW_set, 596
set_flag_update_slot_number, 596
set_rf_link, 596
string_decoder, 596
glonass_gnav_navigation_message.h, 1967, 1968
GLONASS_GNAV_PREAMBLE
core_system_parameters, 132
GLONASS_GNAV_PREAMBLE_DURATION_S
core_system_parameters, 192
GLONASS_GNAV_PREAMBLE_LENGTH_BITS
core_system_parameters, 192
GLONASS_GNAV_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 192
GLONASS_GNAV_PREAMBLE_PERIOD_BITS
core_system_parameters, 192
GLONASS_GNAV_PREAMBLE_PERIOD_SYMBOLS
core_system_parameters, 192
GLONASS_GNAV_PREAMBLE_SAMPLES
core_system_parameters, 132
GLONASS_GNAV_PREAMBLE_STR
core_system_parameters, 193
GLONASS_GNAV_SECONDS_PER_WEEK
core_system_parameters, 193
GLONASS_GNAV_STRING_BIBINARY_WITH_PREABLE
core_system_parameters, 193
GLONASS_GNAV_STRING_BITS
core_system_parameters, 193
GLONASS_GNAV_STRING_SYMBOLS_WITH_PREABLE
core_system_parameters, 193
glonass_gnav_telemetry_decoder_gs, 597
~glonass_gnav_telemetry_decoder_gs, 598
general_work, 598
reset, 598
set_channel, 598
set_satellite, 598
glonass_gnav_telemetry_decoder_gs.h, 1530
glonass_gnav_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
GLONASS_GNAV_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 193
GLONASS_GNAV_TELEMETRY_RATE_SYMBOLS_SECOND
core_system_parameters, 193
GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 193
GLONASS_GNAV_TELEMETRY_SYMBOLS_PER_PREAMBLE_BIT
core_system_parameters, 193
Glonass_Gnav_Utc_Model, 598
announced_leap_second, 600
d_B1, 600
d_B2, 600
d_KP, 600
d_N_4, 600
d_N_A, 600
d_tau_c, 601
d_tau_gps, 601
Glonass_Gnav_Utc_Model, 599
serialize, 600
utc_time, 600
valid, 601
glonass_gnav_utc_model
Rtklib_Solver, 919
glonass_gnav_utc_model.h, 1971
GLONASS_GRAVITY
core_system_parameters, 193
GLONASS_GRAVITY_CORRECTION
core_system_parameters, 193
GLONASS_J2
core_system_parameters, 194
GLONASS_J4
core_system_parameters, 194
GLONASS_J6
core_system_parameters, 194
GLONASS_J8
core_system_parameters, 194
GLONASS_L1_CA_CHIP_PERIOD_S
core_system_parameters, 194
glonass_l1_ca_code_gen_complex
algorithms_libs, 245
glonass_l1_ca_code_gen_complex_sampled
algorithms_libs, 246
glonass_l1_ca_code_gen_float
algorithms_libs, 246
GLONASS_L1_CA_CODE_LENGTH_CHIPS
core_system_parameters, 194
GLONASS_L1_CA_CODE_PERIOD_MS
core_system_parameters, 194
GLONASS_L1_CA_CODE_PERIOD_S
core_system_parameters, 194
GLONASS_L1_CA_CODE_RATE_CPS
core_system_parameters, 194
GLONASS_L1_CA_DFREQ_HZ
core_system_parameters, 194
GLONASS_L1_CA_FREQ_HZ
core_system_parameters, 195
GLONASS_L1_CA_HISTORY_DEEP
core_system_parameters, 195
GLONASS_L1_CA_SYMBOL_PERIOD_S
core_system_parameters, 195

GLONASS_L1_CA_SYMBOL_RATE_BPS
core_system_parameters, 195
GLONASS_L1_L2_CA.h, 1973, 1977
glonass_l1_signal_replica.h, 1147
GLONASS_L2_CA_CHIP_PERIOD_S
core_system_parameters, 195
glonass_l2_ca_code_gen_complex
algorithms_libs, 246
glonass_l2_ca_code_gen_complex_sampled
algorithms_libs, 246
glonass_l2_ca_code_gen_float
algorithms_libs, 246
GLONASS_L2_CA_CODE_LENGTH_CHIPS
core_system_parameters, 195
GLONASS_L2_CA_CODE_PERIOD_MS
core_system_parameters, 195
GLONASS_L2_CA_CODE_PERIOD_S
core_system_parameters, 195
GLONASS_L2_CA_CODE_RATE_CPS
core_system_parameters, 195
GLONASS_L2_CA_DFREQ_HZ
core_system_parameters, 195
GLONASS_L2_CA_FREQ_HZ
core_system_parameters, 196
GLONASS_L2_CA_SYMBOL_PERIOD_S
core_system_parameters, 196
GLONASS_L2_CA_SYMBOL_RATE_BPS
core_system_parameters, 196
glonass_l2_signal_replica.h, 1148
GLONASS_LEAP_SECONDS
core_system_parameters, 196
GLONASS_MOON_ECCENTRICITY
core_system_parameters, 196
GLONASS_MOON_GM
core_system_parameters, 196
GLONASS_MOON_INCLINATION
core_system_parameters, 196
GLONASS_MOON_OMEGA_0
core_system_parameters, 197
GLONASS_MOON_OMEGA_1
core_system_parameters, 197
GLONASS_MOON_Q0
core_system_parameters, 197
GLONASS_MOON_Q1
core_system_parameters, 197
GLONASS_MOON_SEMI_MAJOR_AXIS
core_system_parameters, 197
GLONASS_OMEGA_EARTH_DOT
core_system_parameters, 197
GLONASS_PRN
core_system_parameters, 197
GLONASS_SEMI_MAJOR_AXIS
core_system_parameters, 197
glonass_strict_health
Pvt_Conf, 857
GLONASS_SUN_ECCENTRICITY
core_system_parameters, 197
GLONASS_SUN_GM
core_system_parameters, 197
GLONASS_SUN_OMEGA
core_system_parameters, 198
GLONASS_SUN_Q0
core_system_parameters, 198
GLONASS_SUN_Q1
core_system_parameters, 198
GLONASS_SUN_SEMI_MAJOR_AXIS
core_system_parameters, 198
GLONASS_TAU_0
core_system_parameters, 198
GLONASS_TAU_1
core_system_parameters, 198
GLONASS_U0
core_system_parameters, 198
glot_to_gpst
Glonass_Gnav_Ephemeris, 586
glot_to_utc
Glonass_Gnav_Ephemeris, 587
GME
rtklib_tides.h, 1271
GMM
rtklib_tides.h, 1271
GMS
rtklib_tides.h, 1271
gnmax_signal_source.h, 1423
GnMaxSignalSource, 601
connect, 602
disconnect, 602
get_left_block, 602
get_right_block, 602
item_size, 602
GNSS block interfaces, 83
~SignalSourceInterface, 84
acquisition_impl_interface_sptr, 84
gnss_make_shared, 84
gnss_shared_ptr, 84
SignalSourceInterface, 84
GNSS-SDR, 1
Gnss_Almanac, 603
af0, 605
af1, 605
delta_i, 605
ecc, 605
from_startup_load, 605
Gnss_Almanac, 604
M_0, 605
omega, 605
OMEGA_0, 605
OMEGAdot, 605
predicted_doppler, 604
PRN, 606
satellitePosVelComputation, 604
sqrtA, 606
System, 606
toa, 606
WNa, 606
gnss_almanac.h, 1982

gnss_block_factory.h, 1706, 1707
gnss_block_interface.h, 1639, 1640
Gnss_circular_deque
    algorithms_libs, 246
Gnss_circular_deque< T >, 606
gnss_circular_deque.h, 1149, 1150
Gnss_Crypto, 607
    ~Gnss_Crypto, 608
    clear_public_key, 608
    compute_CMAC_AES, 608
    compute_HMAC_SHA_256, 608
    compute_SHA3_256, 609
    compute_SHA_256, 609
    get_merkle_root, 609
    get_public_key_compressed, 609
    get_public_key_type, 609
    Gnss_Crypto, 608
    have_public_key, 609
    set_merkle_root, 609
    set_public_key, 609
    set_public_key_type, 610
    store_public_key, 610
    verify_signature_ecdsa_p256, 610
    verify_signature_ecdsa_p521, 610
gnss_crypto.h, 1656
Gnss_Ephemeris, 610
    A0, 613
    Adot, 613
    af0, 613
    af1, 614
    af2, 614
    Cic, 614
    Cis, 614
    Crc, 614
    Crs, 614
    Cuc, 614
    Cus, 614
    delta_n, 615
    delta_ndot, 615
    dtr, 615
    ecc, 615
    i_0, 615
    idot, 615
    M_0, 615
    omega, 616
    OMEGA_0, 616
    OMEGAdot, 616
    predicted_doppler, 612
    PRN, 616
    satClkDrift, 616
    satellitePosition, 613
    satpos_X, 616
    satpos_Y, 616
    satpos_Z, 616
    satvel_X, 617
    satvel_Y, 617
    satvel_Z, 617
    sqrtA, 617
    sv_clock_drift, 613
    System, 617
    toc, 617
    toe, 617
    tow, 617
    WN, 618
gnss_ephemeris.h, 1983, 1984
gnss_fft_complex_fwd
    gnss_sdr_fft.h, 1153
gnss_fft_complex_rev
    gnss_sdr_fft.h, 1153
gnss_fft_fwd_make_unique
    gnss_sdr_fft.h, 1153
gnss_fft_fwd_unique_ptr
    gnss_sdr_fft.h, 1153
gnss_fft_rev_make_unique
    gnss_sdr_fft.h, 1153
gnss_fft_rev_unique_ptr
    gnss_sdr_fft.h, 1153
gnss_flowgraph.h, 1708, 1709
gnss_frequencies.h, 1985, 1986
gnss_id
    Bds3_B1c_Bgto, 361
gnss_id_clock_subset
    Galileo_HAS_data, 551
gnss_id_mask
    Galileo_HAS_data, 551
gnss_iod
    Galileo_HAS_data, 551
gnss_make_shared
    GNSS block interfaces, 84
gnss_obs_codes.h, 1987, 1990
GNSS_OMEGA_EARTH_DOT
    core_system_parameters, 198
GNSS_PI
    core_system_parameters, 198
Gnss_Satellite, 618
    ~Gnss_Satellite, 619
    get_block, 619
    get_PRN, 619
    get_rf_link, 619
    get_system, 620
    get_system_short, 620
    Gnss_Satellite, 619
    operator<<, 620
    operator=, 620
    operator==, 620
    update_PRN, 620
    what_block, 620
gnss_satellite.h, 1991
GNSS_SDR_BEAMFORMER_CHANNELS
    input_filter_gr_blocks, 231
gnss_sdr_create_directory.h, 1151, 1152
gnss_sdr_fft.h, 1152, 1154
    gnss_fft_complex_fwd, 1153
    gnss_fft_complex_rev, 1153
    gnss_fft_fwd_make_unique, 1153
    gnss_fft_fwd_unique_ptr, 1153

gnss_fft_rev_make_unique, 1153
gnss_fft_rev_unique_ptr, 1153
gnss_sdr_filesystem.h, 1154, 1155
gnss_sdr_flags, 255
ABSL_DECLARE_FLAG, 256–259
DEFAULT_CARRIER_SMOOTHING_FACTOR, 259
gnss_sdr_flags.h, 1155, 1157
gnss_sdr_fpga_sample_counter, 621
gnss_sdr_fpga_sample_counter.h, 1658
gnss_sdr_fpga_sample_counter_sptr
core_libs, 87
gnss_sdr_log_sink.h, 1659, 1660
gnss_sdr_make_unique.h, 1159
gnss_sdr_sample_counter, 621
gnss_sdr_sample_counter.h, 1661, 1662
gnss_sdr_sample_counter_sptr
core_libs, 87
gnss_sdr_string_literals.h, 1161
Gnss_Sdr_Supl_Client, 622
gal_almanac_map, 627
gal_ephemeris_map, 627
gal_iono, 627
gal_utc, 627
glo_gnav_utc, 627
glonass_gnav_ephemeris_map, 627
gps_acq_map, 627
gps_almanac_map, 627
gps_cnav_ephemeris_map, 627
gps_cnav_utc, 627
gps_ephemeris_map, 627
gps_iono, 627
gps_ref_loc, 627
gps_time, 628
gps_utc, 628
load_cnav_ephemeris_xml, 623
load_cnav_utc_xml, 623
load_ephemeris_xml, 624
load_gal_almanac_xml, 624
load_gal_ephemeris_xml, 624
load_gal_iono_xml, 624
load_gal_utc_xml, 624
load_glo_utc_xml, 624
load_gnav_ephemeris_xml, 624
load_gps_almanac_xml, 624
load_iono_xml, 624
load_ref_location_xml, 624
load_ref_time_xml, 625
load_utc_xml, 625
request, 628
save_cnav_ephemeris_map_xml, 625
save_cnav_utc_xml, 625
save_ephemeris_map_xml, 625
save_gal_almanac_xml, 625
save_gal_ephemeris_map_xml, 625
save_gal_iono_xml, 625
save_gal_utc_xml, 625
save_glo_utc_xml, 626
save_gnav_ephemeris_map_xml, 626
save_gps_almanac_xml, 626
save_iono_xml, 626
save_ref_location_xml, 626
save_ref_time_xml, 626
save_utc_xml, 626
server_name, 628
server_port, 628
gnss_sdr_supl_client.h, 1663
gnss_sdr_time_counter, 628
gnss_sdr_time_counter.h, 1667
gnss_sdr_time_counter_sptr
core_libs, 87
Gnss_Sdr_Timestamp, 629
gnss_sdr_timestamp.h, 1501
Gnss_Sdr_Valve, 629
gnss_sdr_valve.h, 1502, 1503
gnss_shared_ptr
GNSS block interfaces, 84
Gnss_Signal, 630
get_satellite, 630
get_signal_str, 630
operator<<, 631
operator==, 631
gnss_signal.h, 1993
gnss_signal_replica.h, 1161, 1162
Gnss_Synchro, 631
∼Gnss_Synchro, 633
Acq_delay_samples, 634
Acq_doppler_hz, 634
Acq_doppler_step, 634
Acq_samplestamp_samples, 634
Carrier_Doppler_hz, 634
Carrier_phase_rads, 634
Channel_ID, 634
CN0_dB_hz, 634
Code_phase_samples, 635
correlation_length_ms, 635
Flag_carrier_phase_continuous, 635
Flag_cycle_slip, 635
Flag_half_cycle_slip, 635
Flag_PLL_180_deg_phase_locked, 635
Flag_valid_acquisition, 635
Flag_valid_pseudorange, 635
Flag_valid_symbol_output, 635
Flag_valid_word, 636
fs, 636
Gnss_Synchro, 633
interp_TOW_ms, 636
operator=, 633
PRN, 636
Prompt_I, 636
Prompt_Q, 636
Pseudorange_m, 636
RX_time, 636
serialize, 633
Signal, 637
System, 637

TOW_at_current_symbol_ms, 637
Tracking_sample_counter, 637
gnss_synchro.h, 1994
gnss_synchro_monitor, 637
~gnss_synchro_monitor, 638
gnss_synchro_monitor.h, 1691, 1692
gnss_synchro_monitor_sptr
core_monitor, 89
Gnss_Synchro_Udp_Sink, 638
gnss_synchro_udp_sink.h, 1693, 1694
gnss_time.h, 1163, 1164
gnss_tow, 324
add_ms, 324
circular_error_ms, 324
floor_div, 325
floor_to_second_ms, 325
sample_counter_delta, 325
seconds_to_ms, 325
to_ms, 325
week_after_delta, 325
WEEK_MS, 326
WEEK_S, 326
wrap_ms, 325
wrap_s, 325
wrap_s_to_ms, 326
GNSSBlockInterface, 638
connect, 639
disconnect, 640
get_left_block, 640
get_right_block, 640
implementation, 640
item_size, 640
role, 640
start, 640
GNSSFlowgraph, 641
~GNSSFlowgraph, 642
acquisition_manager, 642
apply_action, 642
connect, 642
connected, 642
disconnect, 642
get_pvt, 642
GNSSFlowgraph, 642
MaybeUpdateVisibility, 643
priorize_satellites, 643
running, 643
send_telemetry_msg, 643
set_configuration, 643
start, 643
stop, 643
stop_duplicated_satellite_channels, 643
UpdateVisibilityReference, 643
visibility_aware_search_enabled, 644
wait, 644
GnssSdrLogSink, 644
filename, 644
GnssSdrLogSinkGuard, 645
GnssTime, 645
rx_time, 645
tow_ms, 645
tow_ms_fraction, 645
week, 645
Gps_Acq_Assist, 646
Azimuth, 646
Code_Phase, 646
Code_Phase_int, 647
Code_Phase_window, 647
Doppler0, 647
Doppler1, 647
dopplerUncertainty, 647
Elevation, 647
Gps_Acq_Assist, 646
GPS_Bit_Number, 647
PRN, 647
tow, 647
gps_acq_assist.h, 1997, 1998
gps_acq_map
Gnss_Sdr_Supl_Client, 627
Gps_Almanac, 648
AS_status, 650
get_system, 649
Gps_Almanac, 649
serialize, 649
set_system, 649
SV_health, 650
gps_almanac.h, 1999
gps_almanac_map
Gnss_Sdr_Supl_Client, 627
Rtklib_Solver, 919
GPS_Bit_Number
Gps_Acq_Assist, 647
GPS_CA_PREAMBLE
core_system_parameters, 198
GPS_CA_PREAMBLE_DURATION_MS
core_system_parameters, 199
GPS_CA_PREAMBLE_DURATION_S
core_system_parameters, 199
GPS_CA_PREAMBLE_LENGTH_BITS
core_system_parameters, 199
GPS_CA_PREAMBLE_LENGTH_SYMBOLS
core_system_parameters, 199
GPS_CA_PREAMBLE_SYMBOLS_STR
core_system_parameters, 199
GPS_CA_TELEMETRY_RATE_BITS_SECOND
core_system_parameters, 199
GPS_CA_TELEMETRY_RATE_SYMBOLS_SECOND
core_system_parameters, 199
GPS_CA_TELEMETRY_SYMBOLS_PER_BIT
core_system_parameters, 199
GPS_CNAV.h, 2000, 2003
GPS_CNAV_DATA_PAGE_BITS
core_system_parameters, 199
Gps_CNAV_Eop, 650
delta_ut1_gps, 651
delta_ut1_gps_dot, 651
pm_x, 651

pm_x_dot, 651
pm_y, 651
pm_y_dot, 651
PRN, 651
t_eop, 651
tow, 651
valid, 651
gps_cnav_eop
Rtklib_Solver, 920
gps_cnav_eop.h, 2005, 2006
Gps_CNAV_Ephemeris, 652
alert_flag, 655
antispoofing_flag, 655
delta_A, 655
delta_OMEGAdot, 656
Gps_CNAV_Ephemeris, 655
integrity_status_flag, 656
ISCL1, 656
ISCL2, 656
ISCL5I, 656
ISCL5Q, 656
l2c_phasing_flag, 656
serialize, 655
signal_health, 656
TGD, 656
toe1, 657
toe2, 657
top, 657
URA, 657
URA0, 657
URA1, 657
URA2, 657
URAED, 657
URANED0, 658
URANED1, 658
URANED2, 658
WNop, 658
gps_cnav_ephemeris.h, 2006, 2007
gps_cnav_ephemeris_map
Gnss_Sdr_Supl_Client, 627
Rtklib_Solver, 920
Gps_CNAV_Iono, 658
Gps_CNAV_Iono, 659
gps_cnav_iono
Rtklib_Solver, 920
gps_cnav_iono.h, 2009
Gps_CNAV_Navigation_Message, 659
get_eop, 660
get_ephemeris, 660
get_iono, 660
get_utc_model, 660
Gps_CNAV_Navigation_Message, 660
have_new_eop, 660
have_new_ephemeris, 661
have_new_iono, 661
have_new_utc_model, 661
gps_cnav_navigation_message.h, 2010
gps_cnav_utc
Gnss_Sdr_Supl_Client, 627
Gps_CNAV_Utc_Model, 661
Gps_CNAV_Utc_Model, 662
gps_cnav_utc_model
Rtklib_Solver, 920
gps_cnav_utc_model.h, 2012
Gps_Ephemeris, 662
alert_flag, 666
antispoofing_flag, 666
AODO, 666
code_on_L2, 666
fit_interval_flag, 666
get_system, 665
Gps_Ephemeris, 665
integrity_status_flag, 667
IODC, 667
IODE_SF2, 667
IODE_SF3, 667
L2_P_data_flag, 667
satelliteBlock, 667
serialize, 665
set_system, 666
spare1, 667
spare2, 667
SV_accuracy, 667
SV_health, 668
TGD, 668
gps_ephemeris.h, 2013
gps_ephemeris_map
Gnss_Sdr_Supl_Client, 627
Rtklib_Solver, 920
GPS_F
core_system_parameters, 199
GPS_GM
core_system_parameters, 199
Gps_Iono, 668
alpha0, 669
alpha1, 669
alpha2, 669
alpha3, 670
beta0, 670
beta1, 670
beta2, 670
beta3, 670
Gps_Iono, 669
serialize, 669
valid, 670
gps_iono
Gnss_Sdr_Supl_Client, 627
Rtklib_Solver, 920
gps_iono.h, 2015, 2016
GPS_L1_CA.h, 2017, 2022
GPS_L1_CA_BIT_PERIOD_MS
core_system_parameters, 200
GPS_L1_CA_CHIP_PERIOD_S
core_system_parameters, 200
gps_l1_ca_code_gen_complex
algorithms_libs, 247

gps_l1_ca_code_gen_complex_sampled
algorithms_libs, 247
gps_l1_ca_code_gen_float
algorithms_libs, 247
gps_l1_ca_code_gen_int
algorithms_libs, 247
GPS_L1_CA_CODE_LENGTH_CHIPS
core_system_parameters, 200
GPS_L1_CA_CODE_PERIOD_MS
core_system_parameters, 200
GPS_L1_CA_CODE_PERIOD_S
core_system_parameters, 200
GPS_L1_CA_CODE_RATE_CPS
core_system_parameters, 200
gps_l1_ca_dll_pll_tracking_gpu.h, 1568
Gps_L1_Ca_Dll_Pll_Tracking_GPU_cc, 670
gps_l1_ca_dll_pll_tracking_gpu_cc.h, 1586
gps_l1_ca_dll_pll_tracking_gpu_cc_sptr
tracking_gr_blocks, 317
gps_l1_ca_gaussian_tracking.h, 1570
Gps_L1_Ca_Gaussian_Tracking_cc, 671
gps_l1_ca_gaussian_tracking_cc.h, 1589
gps_l1_ca_gaussian_tracking_cc_sptr
tracking_gr_blocks, 317
GPS_L1_CA_HISTORY_DEEP
core_system_parameters, 200
gps_l1_ca_kf_tracking.h, 1571, 1572
GPS_L1_CA_OPT_ACQ_FS_SPS
core_system_parameters, 200
gps_l1_ca_tcp_connector_tracking.h, 1573, 1574
Gps_L1_Ca_Tcp_Connector_Tracking_cc, 672
gps_l1_ca_tcp_connector_tracking_cc.h, 1592, 1593
gps_l1_ca_tcp_connector_tracking_cc_sptr
tracking_gr_blocks, 317
gps_l1_ca_telemetry_decoder_gs, 672
general_work, 673
reset, 673
set_channel, 673
set_satellite, 673
gps_l1_ca_telemetry_decoder_gs.h, 1532, 1533
gps_l1_ca_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
GPS_L1_FREQ_HZ
core_system_parameters, 200
GPS_L2_CNAV_DATA_PAGE_BITS
core_system_parameters, 200
GPS_L2_CNAV_DATA_PAGE_DURATION_S
core_system_parameters, 201
GPS_L2_CNAV_DATA_PAGE_SYMBOLS
core_system_parameters, 201
GPS_L2_FREQ_HZ
core_system_parameters, 201
GPS_L2_L_CODE_LENGTH_CHIPS
core_system_parameters, 201
GPS_L2_L_CODE_RATE_CPS
core_system_parameters, 201
GPS_L2_L_PERIOD_S
core_system_parameters, 201
GPS_L2_M_CODE_LENGTH_CHIPS
core_system_parameters, 201
GPS_L2_M_CODE_LENGTH_CHIPS_L_ZEROED
core_system_parameters, 201
GPS_L2_M_CODE_PERIOD_MS
core_system_parameters, 201
GPS_L2_M_CODE_RATE_CPS
core_system_parameters, 201
GPS_L2_M_CODE_RATE_L_ZEROED_CPS
core_system_parameters, 202
GPS_L2_M_PERIOD_S
core_system_parameters, 202
GPS_L2_SAMPLES_PER_SYMBOL
core_system_parameters, 202
GPS_L2_SYMBOLS_PER_BIT
core_system_parameters, 202
GPS_L2_V27_HISTORY_LENGTH_BITS
telemetry_decoder_libswiftcnav, 314
GPS_L2C.h, 2025, 2026
GPS_L2C_HISTORY_DEEP
core_system_parameters, 202
gps_l2c_m_code_gen_complex
algorithms_libs, 247
gps_l2c_m_code_gen_complex_sampled
algorithms_libs, 247, 248
gps_l2c_m_code_gen_float
algorithms_libs, 248
gps_l2c_m_code_gen_float_cl_zeroed
algorithms_libs, 248
GPS_L2C_M_INIT_REG
core_system_parameters, 202
GPS_L2C_OPT_ACQ_FS_SPS
core_system_parameters, 202
gps_l2c_signal_replica.h, 1164, 1165
gps_l2c_telemetry_decoder_gs, 674
general_work, 675
reset, 675
set_channel, 675
set_satellite, 675
gps_l2c_telemetry_decoder_gs.h, 1534, 1535
gps_l2c_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
GPS_L2C_V27_DECODE_BITS
telemetry_decoder_libswiftcnav, 314
GPS_L2C_V27_DELAY_BITS
telemetry_decoder_libswiftcnav, 314
GPS_L2C_V27_INIT_BITS
telemetry_decoder_libswiftcnav, 314
GPS_L5.h, 2027, 2029
GPS_L5_CNAV_DATA_PAGE_BITS
core_system_parameters, 203
GPS_L5_CNAV_DATA_PAGE_DURATION_S
core_system_parameters, 203
GPS_L5_CNAV_DATA_PAGE_SYMBOLS
core_system_parameters, 203
GPS_L5_FREQ_HZ
core_system_parameters, 203
GPS_L5_HISTORY_DEEP

core_system_parameters, 203
GPS_L5_OPT_ACQ_FS_SPS
core_system_parameters, 203
GPS_L5_SAMPLES_PER_SYMBOL
core_system_parameters, 203
gps_l5_signal_replica.h, 1165, 1166
GPS_L5_SYMBOLS_PER_BIT
core_system_parameters, 203
gps_l5_telemetry_decoder_gs, 675
reset, 676
set_channel, 676
set_satellite, 676
gps_l5_telemetry_decoder_gs.h, 1536, 1537
gps_l5_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
gps_l5i_code_gen_complex
algorithms_libs, 248
gps_l5i_code_gen_complex_sampled
algorithms_libs, 248
gps_l5i_code_gen_float
algorithms_libs, 248
GPS_L5I_CODE_LENGTH_CHIPS
core_system_parameters, 203
GPS_L5I_CODE_RATE_CPS
core_system_parameters, 203
GPS_L5I_INIT_REG
core_system_parameters, 203
GPS_L5I_NH_CODE
core_system_parameters, 204
GPS_L5I_NH_CODE_LENGTH
core_system_parameters, 204
GPS_L5I_NH_CODE_STR
core_system_parameters, 204
GPS_L5I_PERIOD_MS
core_system_parameters, 204
GPS_L5I_PERIOD_S
core_system_parameters, 204
GPS_L5I_SYMBOL_PERIOD_MS
core_system_parameters, 204
GPS_L5I_SYMBOL_PERIOD_S
core_system_parameters, 204
gps_l5q_code_gen_complex
algorithms_libs, 249
gps_l5q_code_gen_complex_sampled
algorithms_libs, 249
gps_l5q_code_gen_float
algorithms_libs, 249
GPS_L5Q_CODE_LENGTH_CHIPS
core_system_parameters, 204
GPS_L5Q_CODE_RATE_CPS
core_system_parameters, 204
GPS_L5Q_INIT_REG
core_system_parameters, 204
GPS_L5Q_NH_CODE
core_system_parameters, 204
GPS_L5Q_NH_CODE_LENGTH
core_system_parameters, 205
GPS_L5Q_NH_CODE_STR
core_system_parameters, 205
GPS_L5Q_PERIOD_S
core_system_parameters, 205
GPS_LNAV_DATA_ID
core_system_parameters, 205
Gps_Navigation_Message, 676
get_almanac, 677
get_almanac_config_code, 677
get_almanac_health, 678
get_ephemeris, 678
get_flag_iono_valid, 678
get_flag_utc_model_valid, 678
get_GPS_week, 678
get_iono, 678
get_satellite_PRN, 678
get_TOW, 678
get_utc_model, 678
Gps_Navigation_Message, 677
set_channel, 678
set_satellite_PRN, 679
subframe_decoder, 679
utc_time, 679
gps_navigation_message.h, 2031
gps_previous_ephemeris_map
Rtklib_Solver, 920
gps_ref_loc
Gnss_Sdr_Supl_Client, 627
gps_ref_week_from_config
core_system_parameters, 134
gps_satellites
sdr_gnss_packet_t, 942
gps_sdr_signal_replica.h, 1167
GPS_SUBFRAME_BITS
core_system_parameters, 205
GPS_SUBFRAME_LENGTH
core_system_parameters, 205
GPS_SUBFRAME_MS
core_system_parameters, 205
GPS_SUBFRAME_SECONDS
core_system_parameters, 205
GPS_SV_CONFIG_AS_FLAG_MASK
core_system_parameters, 205
GPS_SV_CONFIG_BLOCK_IIF
core_system_parameters, 205
GPS_SV_CONFIG_BLOCK_IIR_M
core_system_parameters, 206
GPS_SV_CONFIG_CODE_MASK
core_system_parameters, 206
GPS_SV_CONFIG_UNKNOWN
core_system_parameters, 206
gps_time
Gnss_Sdr_Supl_Client, 628
gps_utc
Gnss_Sdr_Supl_Client, 628
Gps_Utc_Model, 679
A0, 680
A1, 680
A2, 680

DeltaT_LS, 680
DeltaT_LSF, 681
DN, 681
Gps_Utc_Model, 680
serialize, 680
tot, 681
valid, 681
WN_LSF, 681
WN_T, 681
gps_utc_model
Rtklib_Solver, 920
gps_utc_model.h, 2035, 2036
gps_week_rollover.h, 2037
GPS_WORD_BITS
core_system_parameters, 206
GPS_WORD_LENGTH
core_system_parameters, 206
gpsglob
exterr_t, 480
GpsL1CaDllPllTrackingGPU, 681
connect, 682
disconnect, 682
get_left_block, 682
get_right_block, 682
implementation, 682
item_size, 683
role, 683
set_channel, 683
set_gnss_synchro, 683
start_tracking, 683
stop_tracking, 683
GpsL1CaGaussianTracking, 683
connect, 684
disconnect, 684
get_left_block, 684
get_right_block, 685
implementation, 685
item_size, 685
role, 685
set_channel, 685
set_gnss_synchro, 685
start_tracking, 685
stop_tracking, 685
GpsL1CaKfTracking, 686
connect, 686
disconnect, 686
get_left_block, 687
get_right_block, 687
implementation, 687
item_size, 687
role, 687
set_channel, 687
set_gnss_synchro, 687
start_tracking, 687
stop_tracking, 687
GpsL1CaTcpConnectorTracking, 688
connect, 689
disconnect, 689
get_left_block, 689
get_right_block, 689
implementation, 689
item_size, 689
role, 689
set_channel, 689
set_gnss_synchro, 689
start_tracking, 689
stop_tracking, 690
GPU_Complex, 690
GPU_Complex, 690
i, 691
magnitude2, 690
multiply_acc, 690
operator+, 691
operator+=, 691
operator∗, 691
r, 691
GPU_Complex_Short, 691
GPU_Complex_Short, 692
i, 692
magnitude2, 692
operator+, 692
operator∗, 692
r, 692
gpx_output_enabled
Pvt_Conf, 857
gpx_output_path
Pvt_Conf, 857
Gpx_Printer, 692
gpx_printer.h, 1310
gpx_rate_ms
Pvt_Conf, 857
Gr_Complex_Ip_Packet_Source, 693
sptr, 693
gr_complex_ip_packet_source.h, 1466
Gravity_ECEF
algorithms_libs, 249
great_circle_distance
algorithms_libs, 249
grid_points
Beidou_Dnav_Iono, 389
grid_valid
Beidou_Dnav_Iono, 389
GridId
CudaPcpsEngine, 435
gss6450_file_signal_source.h, 1424, 1425
GSS6450FileSignalSource, 693
connect, 695
disconnect, 695
filename, 695
get_left_block, 695
get_right_block, 695
getRfChannels, 695
item_size, 695
item_type, 695
repeat, 695
samples, 695

sampling_frequency, 695
GST_START_EPOCH
Osnma_Helper, 796
GST_to_UTC_time
Galileo_Utc_Model, 570
gtime_t, 696
sec, 696
time, 696

half
ssat_t, 981
half_cyc_tag, 696
corr, 696
freq, 696
next, 697
sat, 697
te, 697
ts, 697
valid, 697
HALF_PI
core_system_parameters, 206
halfc
raw_t, 875
has_base_position
Ntrip_Rtcm_Snapshot, 783
has_beidou
Signal_Enabled_Flags, 961
HAS_bias_corrections, 697
bias, 697
iod_set_id, 697
mask_id, 697
valid_until, 698
has_clock_correction_m
eph_t, 473
HAS_clock_corrections, 698
clock_correction_m, 698
iod, 698
iod_set_id, 698
mask_id, 698
valid_until, 698
has_galileo
Signal_Enabled_Flags, 961
has_glonass
Signal_Enabled_Flags, 962
has_gps
Signal_Enabled_Flags, 962
has_message_string
Galileo_HAS_page, 554
HAS_MSG_CODE_BIAS_LENGTH
core_system_parameters, 206
HAS_MSG_CODE_BIAS_SCALE_FACTOR
core_system_parameters, 206
HAS_MSG_DELTA_CLOCK_CORRECTION_LENGTH
core_system_parameters, 206
HAS_MSG_DELTA_CLOCK_CORRECTION_SUBSET_LENGTH
core_system_parameters, 206
HAS_MSG_DELTA_CLOCK_MULTIPLIER_LENGTH
core_system_parameters, 206
HAS_MSG_DELTA_CLOCK_MULTIPLIER_SUBSET_LENGTH
core_system_parameters, 207
HAS_MSG_DELTA_CLOCK_SCALE_FACTOR
core_system_parameters, 207
HAS_MSG_DELTA_CROSS_TRACK_LENGTH
core_system_parameters, 207
HAS_MSG_DELTA_CROSS_TRACK_SCALE_FACTOR
core_system_parameters, 207
HAS_MSG_DELTA_IN_TRACK_LENGTH
core_system_parameters, 207
HAS_MSG_DELTA_IN_TRACK_SCALE_FACTOR
core_system_parameters, 207
HAS_MSG_DELTA_RADIAL_LENGTH
core_system_parameters, 207
HAS_MSG_DELTA_RADIAL_SCALE_FACTOR
core_system_parameters, 207
HAS_MSG_GALILEO_SYSTEM
core_system_parameters, 207
HAS_MSG_GPS_SYSTEM
core_system_parameters, 207
HAS_MSG_ID_CLOCK_SUBSET_LENGTH
core_system_parameters, 207
HAS_MSG_ID_MASK_LENGTH
core_system_parameters, 207
HAS_MSG_IOD_GAL_LENGTH
core_system_parameters, 208
HAS_MSG_IOD_GPS_LENGTH
core_system_parameters, 208
HAS_MSG_MASK_RESERVED_LENGTH
core_system_parameters, 208
HAS_MSG_NAV_MESSAGE_LENGTH
core_system_parameters, 208
HAS_MSG_NSYS_LENGTH
core_system_parameters, 208
HAS_MSG_NSYS_SUB_LENGTH
core_system_parameters, 208
HAS_MSG_NUMBER_GNSS_IDS
core_system_parameters, 208
HAS_MSG_NUMBER_MASK_IDS
core_system_parameters, 208
HAS_MSG_NUMBER_MAX_TOH
core_system_parameters, 208
HAS_MSG_NUMBER_MESSAGE_IDS
core_system_parameters, 208
HAS_MSG_NUMBER_SATELLITE_IDS
core_system_parameters, 208
HAS_MSG_NUMBER_SIGNAL_MASKS
core_system_parameters, 208
HAS_MSG_PHASE_BIAS_LENGTH
core_system_parameters, 209
HAS_MSG_PHASE_BIAS_SCALE_FACTOR
core_system_parameters, 209
HAS_MSG_PHASE_DISCONTINUITY_INDICATOR_LENGTH
core_system_parameters, 209
HAS_MSG_SATELLITE_MASK_LENGTH
core_system_parameters, 209
HAS_MSG_SIGNAL_MASK_LENGTH
core_system_parameters, 209
HAS_MSG_VALIDITY_INDEX_LENGTH

core_system_parameters, 209
HAS_MSG_WRONG_SYSTEM
core_system_parameters, 209
HAS_obs_corrections, 699
code_bias_m, 699
phase_bias_cycle, 699
phase_bias_discontinuity, 699
phase_discontinuity_indicator, 699
signal, 699
has_observations
Ntrip_Rtcm_Snapshot, 783
HAS_orbit_corrections, 699
cross_track_m, 700
in_track_m, 700
iod, 700
iod_set_id, 700
mask_id, 700
radial_m, 700
valid_until, 700
has_orbit_cross_track_correction_m
eph_t, 473
has_orbit_in_track_correction_m
eph_t, 473
has_orbit_radial_correction_m
eph_t, 473
has_output_file_path
Pvt_Conf, 857
has_precise_ephemeris
Rtklib_Solver, 917
has_qzss
Signal_Enabled_Flags, 962
HAS_SIGNAL_INDEX_TABLE
core_system_parameters, 209
Has_Simple_Printer, 700
has_simple_printer.h, 1311, 1312
has_status
Galileo_HAS_data, 551
Galileo_HAS_page, 554
has_tag_accumulation
OSNMA_NavData, 802
has_validity_interval_at_tow
Galileo_HAS_data, 549
HAS_VALIDITY_INTERVALS
core_system_parameters, 209
hash_function
Osnma_Merkle_Tree_Material, 797
HasSection
INIReader, 712
HasValue
INIReader, 712
have_new_almanac
Beidou_Dnav_Navigation_Message, 392
Glonass_Gnav_Navigation_Message, 595
have_new_differential_corrections
Beidou_Dnav_Navigation_Message, 392
have_new_eop
Gps_CNAV_Navigation_Message, 660
have_new_ephemeris
Beidou_Cnav2_Navigation_Message, 377
Beidou_Dnav_Navigation_Message, 393
Glonass_Gnav_Navigation_Message, 595
Gps_CNAV_Navigation_Message, 661
have_new_HAS_page
Galileo_Cnav_Message, 528
have_new_iono
Beidou_Dnav_Navigation_Message, 393
Gps_CNAV_Navigation_Message, 661
have_new_utc_model
Beidou_Dnav_Navigation_Message, 393
Glonass_Gnav_Navigation_Message, 595
Gps_CNAV_Navigation_Message, 661
have_public_key
Gnss_Crypto, 609
hdop
Monitor_Pvt, 751
header
an_packet_t, 356
Galileo_HAS_data, 551
MACK_message, 740
health
Bds3_B1c_AlmanacMedium, 359
Bds3_B1c_AlmanacReduced, 360
lexeph_t, 733
healthy
Monitor_Pvt::TrackedSatelliteInfo, 1008
height
Monitor_Pvt, 751
sdr_gnss_packet_t, 942
solopt_t, 974
hex_to_bin
Rtcm, 888
hex_to_binary_converter
algorithms_libs, 249
hex_to_binary_string
algorithms_libs, 250
hex_to_int
Rtcm, 888
hex_to_uint
Rtcm, 889
hf
DSM_KROOT_message, 467
hgt
sta_t, 986
hgts
tec_t, 997
high_dyn
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 455
Kf_Conf, 727
HION
algorithms_libs_rtklib, 281
HistogramBitSynchronizer, 701
bins, 702
edge_phase, 702
epochs_until_next_edge, 702
get_epoch_count, 703

get_histogram, 703
get_total_events, 703
HistogramBitSynchronizer, 702
is_edge_epoch, 703
locked, 703
reset, 704
update, 704
HistogramBitSynchronizer::Config, 422
bit_period_ms, 423
Config, 423
dominance_ratio, 423
epoch_ms, 423
min_events_for_lock, 423
min_prompt_mag, 424
runner_up_margin, 424
stable_best_required, 424
tentative_events_required, 424
transition_confidence, 424
transition_window_epochs, 424
use_phase_dot_detector, 425
hkroot
OSNMA_msg, 798
holdamb
rtk_t, 905
host
Ntrip_Rtcm_Client_Config, 779
ntrip_t, 785
hrclk
ssr_t, 984
hs
Bds3_B1c_PageCommon, 363
Beidou_Cnav1_Ephemeris, 372
hybrid_observables.h, 1286, 1287
hybrid_observables_gs, 704
hybrid_observables_gs.h, 1288, 1289
hybrid_observables_gs_sptr
obs_gr_blocks, 303
HybridObservables, 705
connect, 706
disconnect, 706
get_left_block, 706
get_right_block, 706
implementation, 706
item_size, 706
reset, 706
role, 706

i
GPU_Complex, 691
GPU_Complex_Short, 692
i0
alm_t, 352
eph_t, 473
i_0
Gnss_Ephemeris, 615
I_0_2_LSB
core_system_parameters, 210
I_0_LSB
core_system_parameters, 210
I_DOT_2_LSB
core_system_parameters, 210
I_DOT_LSB
core_system_parameters, 210
i_prn
Sbas_Ephemeris, 931
i_satellite_freq_channel
Glonass_Gnav_Almanac, 583
Glonass_Gnav_Ephemeris, 592
i_satellite_slot_number
Glonass_Gnav_Almanac, 583
Glonass_Gnav_Ephemeris, 593
i_sv_ura
Sbas_Ephemeris, 931
i_t0
Sbas_Ephemeris, 931
IB_PPP
algorithms_libs_rtklib, 271
IB_RTK
algorithms_libs_rtklib, 272
ibyte_to_cbyte.h, 1094, 1095
ibyte_to_complex.h, 1096
ibyte_to_cshort.h, 1097, 1098
IbyteToCbyte, 707
connect, 707
disconnect, 707
get_left_block, 707
get_right_block, 708
implementation, 708
item_size, 708
role, 708
IbyteToComplex, 708
connect, 709
disconnect, 709
get_left_block, 709
get_right_block, 709
implementation, 709
item_size, 709
role, 709
IbyteToCshort, 710
connect, 710
disconnect, 710
get_left_block, 710
get_right_block, 710
implementation, 711
item_size, 711
role, 711
IC_PPP
algorithms_libs_rtklib, 272
icpc
raw_t, 875
icpp
raw_t, 875
id
SensorDataConfiguration, 946
SensorDataFileConfiguration, 947
id_counter
Tag, 992

id_type
SensorDataFile, 947
identifier
SensorDataConfiguration, 946
idot
eph_t, 473
Gnss_Ephemeris, 615
igp
sbsion_t, 939
IGPBAND1
rtklib_sbas.h, 1257
IGPBAND2
rtklib_sbas.h, 1257
II_RTK
algorithms_libs_rtklib, 272
IIO_DEFAULTAD936XAPIFIFOSIZE_SAMPLES
ad936x_iio_samples.h, 1492
IIO_INPUTRAMFIFOSIZE
ad936x_iio_samples.h, 1492
IL_RTK
algorithms_libs_rtklib, 272
imag
clFFT_Complex, 414
clFFT_SplitComplex, 415
implementation
ArraySignalConditioner, 357
BeamformerFilter, 366
ByteToShort, 403
Channel, 406
CshortToGrComplex, 433
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
FirFilter, 499
FreqXlatingFirFilter, 519
GalileoE1TcpConnectorTracking, 573
GenSignalSource, 576
GNSSBlockInterface, 640
GpsL1CaDllPllTrackingGPU, 682
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
HybridObservables, 706
IbyteToCbyte, 708
IbyteToComplex, 709
IbyteToCshort, 711
IshortToComplex, 722
IshortToCshort, 723
MmseResamplerConditioner, 747
NotchFilter, 769
NotchFilterLite, 771
Pass_Through, 806
PcpsAcquisitionAdapter, 829
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 835
PulseBlankingFilter, 854
Rtklib_Pvt, 911
SignalConditioner, 965
SignalGenerator, 966
SignalSourceBase, 969
TelemetryDecoderAdapter, 1000
in_memory_configuration.h, 1713, 1714
in_place_possible
kernel_info_t, 724
in_track_m
HAS_orbit_corrections, 700
inav
Galileo_Ephemeris_Store, 539
inav_current_symbol_delay_ms
galileo_tow_utils.h, 1544
INAV_RS_BUFFER_LENGTH
core_system_parameters, 210
INAV_RS_INFO_VECTOR_LENGTH
core_system_parameters, 210
INAV_RS_PARITY_VECTOR_LENGTH
core_system_parameters, 210
INAV_RS_SUBVECTOR_LENGTH
core_system_parameters, 210
inb
stream_t, 988
inbt
stream_t, 988
inc
tled_t, 1003
index
pclk_t, 807
peph_t, 838
ini.h, 1668, 1669
INI_ALLOW_MULTILINE
core_libs, 86
ini_parse
core_libs, 88
INIReader, 711
Get, 712
GetInteger, 712
HasSection, 712
HasValue, 712
INIReader, 712
ParseError, 712
INIReader.h, 1670
init
cnav_v27_part_t, 417
FileSourceBase, 494
Fpga_Acquisition, 508
pcps_acquisition_fpga, 814
init_carrier_freq_rate_sd_hz_s
Kf_Conf, 727
init_carrier_freq_sd_hz
Kf_Conf, 727
init_carrier_phase_sd_rad
Kf_Conf, 728
init_code_phase_sd_chips
Kf_Conf, 728
init_PRN
Galileo_Inav_Message, 557
INIT_ZWD

algorithms_libs_rtklib, 282
initialize
Tlm_CRC_Stats, 1006
Tracking_2nd_DLL_filter, 1009
initialize_secondary_code
Fpga_Multicorrelator_8sc, 514
initlock
algorithms_libs_rtklib, 272
initrst
prcopt_t, 847
InMemoryConfiguration, 712
is_present, 713
property, 713, 714
set_property, 714
Input Filter, 230
input_filter
ArraySignalConditioner, 357
SignalConditioner, 965
input_filter_adapters, 230
input_filter_gr_blocks, 230
beamformer_sptr, 231
GNSS_SDR_BEAMFORMER_CHANNELS, 231
notch_lite_sptr, 231
notch_sptr, 231
pulse_blanking_cc_sptr, 231
inr
stream_t, 988
insert
Galileo_Ephemeris_Store, 539
INT_SWAP_STAT
algorithms_libs_rtklib, 282
INT_SWAP_TRAC
algorithms_libs_rtklib, 282
integrity_status_flag
Beidou_Dnav_Ephemeris, 387
Gps_CNAV_Ephemeris, 656
Gps_Ephemeris, 667
interleaved_byte_to_complex_byte, 714
interleaved_byte_to_complex_byte.h, 1103, 1104
interleaved_byte_to_complex_byte_sptr
data_type_gr_blocks, 229
interleaved_byte_to_complex_short, 715
interleaved_byte_to_complex_short.h, 1104, 1105
interleaved_byte_to_complex_short_sptr
data_type_gr_blocks, 229
interleaved_short_to_complex_short, 715
interleaved_short_to_complex_short.h, 1106
interleaved_short_to_complex_short_sptr
data_type_gr_blocks, 229
interp_TOW_ms
Gnss_Synchro, 636
interpolate_observable
rtklib_pvt_gs, 913
intpref
prcopt_t, 847
inv
beidou_ldpc.h, 1776
invert
cnav_v27_part_t, 417
iod
dgps_t, 439
HAS_clock_corrections, 698
HAS_orbit_corrections, 700
msm_h_t, 754
raw_t, 876
ssr_t, 984
IOD_a_10
Galileo_Almanac_Helper, 526
IOD_a_7
Galileo_Almanac_Helper, 526
IOD_a_8
Galileo_Almanac_Helper, 526
IOD_a_9
Galileo_Almanac_Helper, 526
IOD_ephemeris
Galileo_Ephemeris, 537
IOD_nav
Galileo_Ephemeris, 538
iod_set_id
HAS_bias_corrections, 697
HAS_clock_corrections, 698
HAS_orbit_corrections, 700
mt1_header, 755
IODa
Galileo_Almanac, 522
IODC
Beidou_Cnav1_Ephemeris, 372
Gps_Ephemeris, 667
iodc
eph_t, 473
iodcrc
ssr_t, 984
IODE
Beidou_Cnav1_Ephemeris, 372
iode
eph_t, 473
geph_t, 579
sbslcorr_t, 939
ssr_t, 984
IODE_SF2
Gps_Ephemeris, 667
IODE_SF3
Gps_Ephemeris, 667
iodev
signal_source_libs, 310
iodf
sbsfcorr_t, 936
iodi
sbsion_t, 939
iodp
sbssat_t, 941
ion
stec_t, 987
ion_bdgim
nav_t, 763
ion_bdgim_valid

nav_t, 763
ion_cmp
nav_t, 763
ion_gal
nav_t, 763
ion_gps
nav_t, 763
ion_gsms.h, 1467, 1468
ion_gsms_chunk_data.h, 1504, 1505
bits_to_item_size, 1505
with_word_type, 1505
ion_gsms_chunk_unpacking_ctx.h, 1508
ion_gsms_signal_source.h, 1427
ion_gsms_stream_encodings.h, 1509, 1513
encoding_from_string, 1510
five_bit_look_up, 1511
four_bit_look_up, 1511
FP, 1511
MS, 1511
MSA, 1511
OB, 1511
OBA, 1511
OG, 1512
OGA, 1512
SIGN, 1512
SM, 1512
SMA, 1512
StreamEncoding, 1510
TC, 1512
TCA, 1512
three_bit_look_up, 1512
two_bit_look_up, 1512
UNKNOWN, 1513
ion_irn
nav_t, 763
ion_qzs
nav_t, 763
IONGSMSChunkData, 716
IONGSMSChunkUnpackingCtx
IONGSMSChunkUnpackingCtx< WT >, 717
IONGSMSChunkUnpackingCtx< WT >, 717
bit_offset_, 717
data_, 717
IONGSMSChunkUnpackingCtx, 717
read_bits, 717
read_lsb_first_, 718
shift_padding, 717
word_bitsize_, 718
word_count_, 718
IONGSMSFileSource, 718
sptr, 719
IONGSMSFileSource::SegmentDescriptor, 944
block, 945
block_extends_to_eof, 945
block_start_offset, 945
data_filepath, 945
IONGSMSSignalSource, 719
connect, 720
disconnect, 720
get_left_block, 720
get_right_block, 720
getRfChannels, 720
item_size, 720
ionoopt
prcopt_t, 847
IONOOPT_BRDC
algorithms_libs_rtklib, 282
IONOOPT_EST
algorithms_libs_rtklib, 282
IONOOPT_IFLC
algorithms_libs_rtklib, 282
IONOOPT_LEX
algorithms_libs_rtklib, 282
IONOOPT_OFF
algorithms_libs_rtklib, 282
IONOOPT_QZS
algorithms_libs_rtklib, 282
IONOOPT_SBAS
algorithms_libs_rtklib, 282
IONOOPT_STEC
algorithms_libs_rtklib, 283
IONOOPT_TEC
algorithms_libs_rtklib, 283
ionosphere_corrections
Sbas_Rtklib_Corrections, 935
ionospheric_grid_valid
Beidou_Dnav_Differential_Corrections, 382
is_alert_page
Galileo_Inav_Message, 557
is_clock_do_not_use
Galileo_HAS_data, 549
is_clock_unavailable
Galileo_HAS_data, 549
is_code_bias_unavailable
Galileo_HAS_data, 549
is_connected
pps_tcp_rx, 844
is_edge_epoch
HistogramBitSynchronizer, 703
is_flag_TOW_set
Glonass_Gnav_Navigation_Message, 595
is_HAS_in_test_mode
Galileo_Cnav_Message, 529
is_HAS_page_dummy
Galileo_Cnav_Message, 529
is_orbit_along_cross_unavailable
Galileo_HAS_data, 549
is_orbit_radial_unavailable
Galileo_HAS_data, 549
is_phase_bias_unavailable
Galileo_HAS_data, 550
is_present
FileConfiguration, 486
InMemoryConfiguration, 713
is_rinex_header_written
Rinex_Printer, 882

is_server_running
Rtcm, 889
is_TOW0_set
Galileo_Inav_Message, 557
is_TOW1_set
Galileo_Fnav_Message, 541
is_TOW2_set
Galileo_Fnav_Message, 541
is_TOW3_set
Galileo_Fnav_Message, 541
is_TOW4_set
Galileo_Fnav_Message, 541
is_TOW5_set
Galileo_Inav_Message, 557
is_TOW6_set
Galileo_Inav_Message, 557
is_valid
CudaPcpsEngine, 436
is_valid_at
Galileo_Reduced_CED, 565
isc
eph_t, 473
lexeph_t, 734
ISC_B1Cd
Beidou_Cnav1_Ephemeris, 372
ISC_B2ad
Beidou_Cnav1_Ephemeris, 372
ISCL1
Gps_CNAV_Ephemeris, 656
ISCL2
Gps_CNAV_Ephemeris, 656
ISCL5I
Gps_CNAV_Ephemeris, 656
ISCL5Q
Gps_CNAV_Ephemeris, 656
ishort_to_complex.h, 1099
ishort_to_cshort.h, 1100, 1101
IshortToComplex, 721
connect, 721
disconnect, 721
get_left_block, 721
get_right_block, 721
implementation, 722
item_size, 722
role, 722
IshortToCshort, 722
connect, 723
disconnect, 723
get_left_block, 723
get_right_block, 723
implementation, 723
item_size, 723
role, 723
IT_PPP
algorithms_libs_rtklib, 272
IT_RTK
algorithms_libs_rtklib, 273
it_size
Acq_Conf, 334
item_size
Ad936xCustomSignalSource, 347
Adrv9361z7035SignalSourceFPGA, 348
ArraySignalConditioner, 358
BeamformerFilter, 367
BladerfSignalSource, 401
ByteToShort, 403
Channel, 406
CshortToGrComplex, 433
CustomUDPSignalSource, 438
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 461
DllPllTrackingAdapterFpga, 463
DMASignalSourceFPGA, 465
Evk1029SignalSource, 477
FifoSignalSource, 483
FileSourceBase, 494
FirFilter, 499
FlexibandSignalSource, 501
Fmcomms2SignalSource, 502
Fmcomms5SignalSourceFPGA, 504
FreqXlatingFirFilter, 519
GalileoE1TcpConnectorTracking, 573
GenSignalSource, 576
GnMaxSignalSource, 602
GNSSBlockInterface, 640
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
GSS6450FileSignalSource, 695
HybridObservables, 706
IbyteToCbyte, 708
IbyteToComplex, 709
IbyteToCshort, 711
IONGSMSSignalSource, 720
IshortToComplex, 722
IshortToCshort, 723
LabsatSignalSource, 732
LimesdrSignalSource, 737
MAX2771EVKITSignalSourceFPGA, 746
MmseResamplerConditioner, 747
MultichannelFileSignalSource, 757
NotchFilter, 769
NotchFilterLite, 771
OsmosdrSignalSource, 793
Pass_Through, 807
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 835
PlutosdrSignalSource, 840
PocketSdrSignalSource, 842
PulseBlankingFilter, 854
RawArraySignalSource, 879
Rtklib_Pvt, 911
RtlTcpSignalSource, 928
SignalConditioner, 965

SignalGenerator, 967
TelemetryDecoderAdapter, 1000
UhdSignalSource, 1021
ZmqSignalSource, 1032
item_type
Acq_Conf, 334
Dll_Pll_Conf, 449
FileSourceBase, 494
GSS6450FileSignalSource, 695
Kf_Conf, 728
MultichannelFileSignalSource, 757
Pass_Through, 807
item_type_converter_t
algorithms_libs, 238
item_type_helpers.h, 1168, 1169
item_type_is_complex
algorithms_libs, 250
item_type_size
algorithms_libs, 250
item_type_valid
algorithms_libs, 250
itemTypeToSize
FileSourceBase, 494
FourBitCpxFileSignalSource, 506
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
SpirFileSignalSource, 980
TwoBitCpxFileSignalSource, 1016
TwoBitPackedFileSignalSource, 1019
itn
DSM_PKR_message, 469
itrf
Ntrip_Rtcm_Snapshot, 783
sta_t, 986

jerk
lexeph_t, 734

kernel
kernel_info_t, 724
kernel_dir_t
fft_internal.h, 1178
kernel_info
cl_fft_plan, 412
kernel_info_t, 723
dir, 724
in_place_possible, 724
kernel, 724
kernel_name, 724
lmem_size, 724
next, 724
num_workgroups, 724
num_workitems_per_workgroup, 724
num_xforms_per_workgroup, 724
kernel_name
kernel_info_t, 724
kernel_string
cl_fft_plan, 412
key
MACK_message, 740
key_type
Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
Osnma_Public_Key_Material, 804
Kf_Conf, 725
bit_synchronization_time_limit_s, 726
carrier_disc_sd_rads, 726
carrier_freq_rate_sd_hz_s, 726
carrier_freq_sd_hz, 726
carrier_lock_test_smoother_alpha, 726
carrier_lock_test_smoother_samples, 726
carrier_lock_th, 726
carrier_phase_sd_rad, 726
cn0_min, 726
cn0_samples, 726
cn0_smoother_alpha, 726
cn0_smoother_samples, 726
code_disc_sd_chips, 726
code_phase_sd_chips, 727
dump, 727
dump_filename, 727
dump_mat, 727
early_late_space_chips, 727
early_late_space_narrow_chips, 727
enable_doppler_correction, 727
extend_correlation_symbols, 727
fs_in, 727
high_dyn, 727
init_carrier_freq_rate_sd_hz_s, 727
init_carrier_freq_sd_hz, 727
init_carrier_phase_sd_rad, 728
init_code_phase_sd_chips, 728
item_type, 728
max_carrier_lock_fail, 728
max_code_lock_fail, 728
pull_in_time_s, 728
signal, 728
slope, 728
smoother_length, 728
spc, 728
system, 728
track_pilot, 728
vector_length, 729
very_early_late_space_chips, 729
very_early_late_space_narrow_chips, 729
y_intercept, 729
kf_conf.h, 1619
kf_tracking, 729
kf_tracking.h, 1595
kf_use_imu_vel
Pvt_Conf, 857
kFieldSize
beidou_ldpc.h, 1777
kml_output_enabled
Pvt_Conf, 857
kml_output_path
Pvt_Conf, 858

Kml_Printer, 730
kml_printer.h, 1313
kml_rate_ms
    Pvt_Conf, 858
kOrder
    beidou_ldpc.h, 1777
kroot
    DSM_KROOT_message, 467
ks
    DSM_KROOT_message, 467
kZero
    beidou_ldpc.h, 1777

L
    obsd_t, 789
L1LnavSystem
    telemetry_decoder_gr_blocks, 313
L2_P_data_flag
    Gps_Ephemeris, 667
l2c_phasing_flag
    Gps_CNAV_Ephemeris, 656
labsat23_source, 730
labsat23_source.h, 1469, 1470
labsat23_source_sptr
    signal_source_gr_blocks, 308
labsat_signal_source.h, 1429, 1430
LabsatSignalSource, 731
    connect, 732
    disconnect, 732
    get_left_block, 732
    get_right_block, 732
    getRfChannels, 732
    item_size, 732
lam
    nav_t, 763
LAM_CARR
    algorithms_libs_rtklib, 283
lambda0red
    Galileo_Reduced_CED, 566
last_batch_size
    cl_fft_plan, 412
last_crc_ok
    Beidou_Cnav2_Navigation_Message, 377
last_error
    CudaPcpsEngine, 436
last_frame_prn
    Beidou_Cnav2_Navigation_Message, 378
last_mes_type
    Beidou_Cnav2_Navigation_Message, 378
last_sow
    Beidou_Cnav2_Navigation_Message, 378
lat
    Agnss_Ref_Location, 349
    sbsigp_t, 937
latitude
    Monitor_Pvt, 751
    sdr_gnss_packet_t, 942
lats
    tec_t, 997
LC
    ambc_t, 354
lcorr
    sbssatp_t, 941
LCv
    ambc_t, 354
leaf_index
    Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
leaps
    nav_t, 763
len
    raw_t, 876
    rtcm_t, 902
length
    fpga_dma-proxy.h, 1495
length_bits
    Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
lex_t, 732
    msgs, 733
    n, 733
    nmax, 733
lexeph
    nav_t, 763
lexeph_t, 733
    acc, 733
    af0, 733
    af1, 733
    health, 733
    isc, 734
    jerk, 734
    pos, 734
    sat, 734
    tgd, 734
    toe, 734
    tof, 734
    ura, 734
    vel, 734
lexion
    nav_t, 763
lexion_t, 734
    coef, 735
    pos0, 735
    t0, 735
    tspan, 735
lexmsg
    raw_t, 876
lexmsg_t, 735
    alert, 735
    msg, 735
    prn, 735
    snr, 735
    stat, 736
    ttt, 736
    type, 736
limesdr_signal_source.h, 1431
LimesdrSignalSource, 736

connect, 737
disconnect, 737
get_left_block, 737
get_right_block, 737
item_size, 737
LLI
obsd_t, 789
lltime
rtcm_t, 902
lmem_size
kernel_info_t, 724
LnavSystem
core_system_parameters, 133
lo_hz
stream_cfg, 988
load_cnav_ephemeris_xml
Gnss_Sdr_Supl_Client, 623
load_cnav_utc_xml
Gnss_Sdr_Supl_Client, 623
load_ephemeris_xml
Gnss_Sdr_Supl_Client, 624
load_gal_almanac_xml
Gnss_Sdr_Supl_Client, 624
load_gal_ephemeris_xml
Gnss_Sdr_Supl_Client, 624
load_gal_iono_xml
Gnss_Sdr_Supl_Client, 624
load_gal_utc_xml
Gnss_Sdr_Supl_Client, 624
load_glo_utc_xml
Gnss_Sdr_Supl_Client, 624
load_gnav_ephemeris_xml
Gnss_Sdr_Supl_Client, 624
load_gps_almanac_xml
Gnss_Sdr_Supl_Client, 624
load_iono_xml
Gnss_Sdr_Supl_Client, 624
load_ref_location_xml
Gnss_Sdr_Supl_Client, 624
load_ref_time_xml
Gnss_Sdr_Supl_Client, 625
load_utc_xml
Gnss_Sdr_Supl_Client, 625
local
ftp_t, 520
lock
file_t, 484
rtcm_t, 902
rtksvr_t, 923
solstat_t, 976
ssat_t, 981
stream_t, 989
lock_channel
Fpga_Multicorrelator_8sc, 514
lock_detectors.h, 1620, 1621
lock_t
algorithms_libs_rtklib, 273
lock_time
Rtcm, 889, 890
Rtcm_Printer, 900
locked
HistogramBitSynchronizer, 703
lockt
raw_t, 876
lod
erpd_t, 476
LOG_PI
rtklib_ppp.h, 1226
log_rinex_nav_bds_cnav1
Rinex_Printer, 883
log_rinex_nav_bds_cnav2
Rinex_Printer, 883
log_rinex_nav_bds_dnav
Rinex_Printer, 883
log_rinex_nav_gal_nav
Rinex_Printer, 883
log_rinex_nav_glo_gnav
Rinex_Printer, 883
log_rinex_nav_gps_cnav
Rinex_Printer, 883
log_rinex_nav_gps_nav
Rinex_Printer, 884
log_source_timetag
Pvt_Conf, 858
log_source_timetag_file
Pvt_Conf, 858
lon
Agnss_Ref_Location, 349
sbsigp_t, 937
longitude
Monitor_Pvt, 751
sdr_gnss_packet_t, 942
lons
tec_t, 997
LOOPMAX
rtklib_lambda.h, 1218
loss
rtcm_t, 902
Lstd
obsd_t, 789

M
tled_t, 1003
M0
alm_t, 353
eph_t, 473
M0_10
Galileo_Almanac_Helper, 526
M0_10_LSB
core_system_parameters, 210
M0_1_LSB
core_system_parameters, 210
M0_7
Galileo_Almanac_Helper, 526
M0_7_LSB
core_system_parameters, 210
M0_9

Galileo_Almanac_Helper, 526
M0_9_LSB
core_system_parameters, 210
m0_rad
Bds3_B1c_AlmanacMedium, 359
M_0
Gnss_Almanac, 605
Gnss_Ephemeris, 615
M_0_LSB
core_system_parameters, 211
mack
OSNMA_msg, 798
MACK_header, 738
cop, 738
macseq, 738
macseq_valid, 738
tag0, 738
tag0_valid, 738
Mack_lookup, 738
Mack_lookup, 739
msg, 739
nt, 739
sequence1, 739
sequence2, 739
MACK_message, 739
allowed_adkds, 740
header, 740
key, 740
nmas, 740
PRNa, 740
tag_and_info, 740
TOW, 740
WN, 740
MACK_tag_and_info, 740
counter, 741
tag, 741
tag_info, 741
valid, 741
MACK_tag_info, 741
ADKD, 741
cop, 741
PRN_d, 741
maclt
DSM_KROOT_message, 467
macseq
MACK_header, 738
macseq_valid
MACK_header, 738
mag
acquisition_impl_interface, 340
AcquisitionInterface, 342
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 531
galileo_pcps_8ms_acquisition_cc, 562
pcps_acquisition, 809
pcps_acquisition_fine_doppler_cc, 812
pcps_acquisition_fpga, 814
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 819
pcps_opencl_acquisition_cc, 822
pcps_quicksync_acquisition_cc, 824
pcps_tong_acquisition_cc, 827
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 835
magnitude2
GPU_Complex, 690
GPU_Complex_Short, 692
MAIN_GRID
CudaPcpsEngine, 435
make_2_steps
Acq_Conf, 334
Acq_Conf_Fpga, 338
make_ntrip_rtcm_client_config
pvt_libs, 300
make_vector_converter
algorithms_libs, 250
marker
sta_t, 986
mask
snrmask_t, 971
mask_flag
mt1_header, 755
mask_id
HAS_bias_corrections, 697
HAS_clock_corrections, 698
HAS_orbit_corrections, 700
mt1_header, 755
MATH_CONSTANTS.h, 2038, 2041
matlab_type_traits< double >, 742
class_type, 742
data_type, 742
matlab_type_traits< float >, 742
class_type, 742
data_type, 742
matlab_type_traits< int32_t >, 743
class_type, 743
data_type, 743
matlab_type_traits< int64_t >, 743
class_type, 743
data_type, 743
matlab_type_traits< T >, 742
matlab_type_traits< uint32_t >, 744
class_type, 744
data_type, 744
matlab_type_traits< uint64_t >, 744
class_type, 744
data_type, 744
matlab_type_traits< uint8_t >, 744
class_type, 745
data_type, 745
matlab_writter_helper.h, 1170, 1171
write_matlab_var, 1170
MAX
telemetry_decoder_libswiftcnav, 314
max2771_evkit_signal_source_fpga.h, 1432, 1433

MAX2771EVKITSignalSourceFPGA, 745
connect, 746
disconnect, 746
get_left_block, 746
get_right_block, 746
item_size, 746
max_age_s
Ntrip_Rtcm_Client_Config, 779
max_carrier_lock_fail
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 455
Kf_Conf, 728
max_code_lock_fail
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 455
Kf_Conf, 728
max_dwells
Acq_Conf, 334
max_localmem_fft_size
cl_fft_plan, 412
max_num_acqs
Acq_Conf_Fpga, 338
max_obs_block_rx_clock_offset_ms
Pvt_Conf, 858
max_radix
cl_fft_plan, 412
MAX_SECONDS_REMEMBERING_MASK_IOD
core_system_parameters, 211
MAX_SECONDS_REMEMBERING_MID
core_system_parameters, 211
MAX_TOA_DELAY_MS
core_system_parameters, 211
max_work_item_per_workgroup
cl_fft_plan, 412
MAXAC
algorithms_libs_rtklib, 283
MAXANT
algorithms_libs_rtklib, 283
maxaveep
prcopt_t, 847
MAXBAND
algorithms_libs_rtklib, 283
MAXCLI
algorithms_libs_rtklib, 283
rtklib_stream.h, 1266
MAXCODE
core_system_parameters, 211
MAXDTE
rtklib_preceph.h, 1230
MAXDTOE
algorithms_libs_rtklib, 283
MAXDTOE_BDS
algorithms_libs_rtklib, 283
MAXDTOE_GAL
algorithms_libs_rtklib, 283
MAXDTOE_GLO
algorithms_libs_rtklib, 284
MAXDTOE_QZS
algorithms_libs_rtklib, 284
MAXDTOE_S
algorithms_libs_rtklib, 284
MAXDTOE_SBS
algorithms_libs_rtklib, 284
MAXERRMSG
algorithms_libs_rtklib, 284
MAXEXFILE
algorithms_libs_rtklib, 284
MAXFREQ
algorithms_libs_rtklib, 284
MAXGDOP
algorithms_libs_rtklib, 284
maxgdop
prcopt_t, 847
maxinno
prcopt_t, 847
MAXITR
rtklib_pntpos.h, 1221
MAXLEAPS
algorithms_libs_rtklib, 284
MAXNGEO
algorithms_libs_rtklib, 284
MAXNIGP
algorithms_libs_rtklib, 285
MAXOBS
algorithms_libs_rtklib, 285
MAXOBSBUF
algorithms_libs_rtklib, 285
MAXOBSTYPE
algorithms_libs_rtklib, 285
maxout
prcopt_t, 848
MAXPRNBDS
algorithms_libs_rtklib, 285
MAXPRNGAL
algorithms_libs_rtklib, 285
MAXPRNGLO
algorithms_libs_rtklib, 285
MAXPRNGPS
algorithms_libs_rtklib, 285
MAXPRNIRN
algorithms_libs_rtklib, 285
MAXPRNLEO
algorithms_libs_rtklib, 285
MAXPRNQZS
algorithms_libs_rtklib, 286
MAXPRNQZS_S
algorithms_libs_rtklib, 286
MAXPRNSBS
algorithms_libs_rtklib, 286
MAXRAWLEN
algorithms_libs_rtklib, 286
MAXRCV
algorithms_libs_rtklib, 286
MAXSAT
algorithms_libs_rtklib, 286
MAXSBSAGEF

algorithms_libs_rtklib, 286
MAXSBSAGEL
algorithms_libs_rtklib, 286
MAXSBSMSG
algorithms_libs_rtklib, 286
MAXSBSURA
algorithms_libs_rtklib, 286
MAXSOLBUF
algorithms_libs_rtklib, 287
MAXSOLMSG
algorithms_libs_rtklib, 287
MAXSOLQ
algorithms_libs_rtklib, 287
maxsolstd
solopt_t, 974
MAXSTA
algorithms_libs_rtklib, 287
MAXSTATMSG
algorithms_libs_rtklib, 287
rtklib_stream.h, 1266
MAXSTRMSG
algorithms_libs_rtklib, 287
MAXSTRPATH
algorithms_libs_rtklib, 287
MAXSTRRTK
algorithms_libs_rtklib, 287
maxtdiff
prcopt_t, 848
MaybeUpdateVisibility
GNSSFlowgraph, 643
me
ControlThread, 429
measures_ecef_pos_sd_m
Pvt_Conf, 858
measures_ecef_vel_sd_ms
Pvt_Conf, 858
medium_almanac
Bds3_B1c_PageData, 364
message
Ntrip_Rtcm_Client_Status, 781
message_id
Galileo_HAS_data, 551
Galileo_HAS_page, 554
message_lock
cnav_v27_part_t, 417
message_page_id
Galileo_HAS_page, 554
message_size
Galileo_HAS_page, 554
message_type
Galileo_HAS_page, 554
Sbas_Raw_Message, 934
metrics1
v27_t, 1028
metrics2
v27_t, 1028
mf
DSM_KROOT_message, 467
microseconds
Agnss_Ref_Time, 350
sdr_gnss_packet_t, 942
mid
DSM_PKR_message, 469
MIN
telemetry_decoder_libswiftcnav, 314
MIN_ARC_GAP
rtklib_ppp.h, 1226
MIN_EL
rtklib_ionex.h, 1215
min_events_for_lock
HistogramBitSynchronizer::Config, 423
MIN_HGT
rtklib_ionex.h, 1215
min_mem_coalesce_width
cl_fft_plan, 412
MIN_PPP
rtklib_ppp.h, 1225
min_prompt_mag
HistogramBitSynchronizer::Config, 424
mindropsats
prcopt_t, 848
minfix
prcopt_t, 848
minfixsats
prcopt_t, 848
minholdsats
prcopt_t, 848
minlock
prcopt_t, 848
MINPRNBDS
algorithms_libs_rtklib, 287
MINPRNGAL
algorithms_libs_rtklib, 287
MINPRNGLO
algorithms_libs_rtklib, 288
MINPRNGPS
algorithms_libs_rtklib, 288
MINPRNIRN
algorithms_libs_rtklib, 288
MINPRNLEO
algorithms_libs_rtklib, 288
MINPRNQZS
algorithms_libs_rtklib, 288
MINPRNQZS_S
algorithms_libs_rtklib, 288
MINPRNSBS
algorithms_libs_rtklib, 288
mjd
erpd_t, 476
mmse_resampler_conditioner.h, 1384, 1385
MmseResamplerConditioner, 746
connect, 747
disconnect, 747
get_left_block, 747
get_right_block, 747
implementation, 747

item_size, 747
role, 747
mntpnt
ntrip_t, 785
mode
file_t, 484
prcopt_t, 848
stream_t, 989
modear
prcopt_t, 848
ModelFunction, 748
ModelFunction, 748
moni
rtksvr_t, 923
monitor_enabled
Pvt_Conf, 858
monitor_ephemeris_enabled
Pvt_Conf, 858
Monitor_Ephemeris_Udp_Sink, 748
monitor_ephemeris_udp_sink.h, 1314
Monitor_Pvt, 748
AR_ratio_factor, 750
AR_ratio_threshold, 750
cog, 750
cov_xx, 750
cov_xy, 750
cov_yy, 750
cov_yz, 750
cov_zx, 750
cov_zz, 750
galhas_status, 750
gdop, 751
geohash, 751
hdop, 751
height, 751
latitude, 751
longitude, 751
pdop, 751
pos_x, 751
pos_y, 751
pos_z, 751
RX_time, 751
serialize, 750
solution_status, 751
solution_type, 752
TOW_at_current_symbol_ms, 752
tracked_satellites, 752
user_clk_drift_ppm, 752
user_clk_offset, 752
utc_time, 752
valid_sats, 752
vdop, 752
vel_e, 752
vel_n, 752
vel_u, 752
vel_x, 752
vel_y, 753
vel_z, 753
week, 753
monitor_pvt.h, 1315, 1316
Monitor_Pvt::TrackedSatelliteInfo, 1007
azimuth_deg, 1008
combined, 1008
elevation_deg, 1008
healthy, 1008
prn, 1008
serialize, 1008
signal, 1008
system, 1009
used, 1009
Monitor_Pvt_Udp_Sink, 753
monitor_pvt_udp_sink.h, 1318, 1319
monitored
Beidou_Dnav_Iono_Grid_Point, 390
mountpoint
Ntrip_Rtcm_Client_Config, 779
MS
ion_gsms_stream_encodings.h, 1511
ms_per_code
Acq_Conf, 334
MSA
ion_gsms_stream_encodings.h, 1511
msg
lexmsg_t, 735
Mack_lookup, 739
rtcm_t, 903
sbsmsg_t, 940
stream_t, 989
MSG_DISCONN
algorithms_libs_rtklib, 273
msg_handler_osnma
osnma_msg_receiver, 799
msg_id
cnav_msg_t, 416
msgs
lex_t, 733
sbs_t, 936
msgtype
raw_t, 876
rtcm_t, 903
msm_h_t, 753
cellmask, 754
clk_ext, 754
clk_str, 754
iod, 754
nsat, 754
nsig, 754
sats, 754
sigs, 754
smooth, 754
time_s, 754
tint_s, 754
msmtype
rtcm_t, 903
mt1_header, 755
clock_fullset_flag, 755

clock_subset_flag, 755
code_bias_flag, 755
iod_set_id, 755
mask_flag, 755
mask_id, 755
orbit_correction_flag, 755
phase_bias_flag, 755
reserved, 755
toh, 756
mul
beidou_ldpc.h, 1776
multichannel_file_signal_source.h, 1435
MultichannelFileSignalSource, 756
connect, 757
disconnect, 757
filename, 757
get_left_block, 757
get_right_block, 757
item_size, 757
item_type, 757
repeat, 757
samples, 757
sampling_frequency, 758
multiply_acc
GPU_Complex, 690
mw
ssat_t, 981

n
ambc_t, 354
cl_fft_plan, 413
erp_t, 475
lex_t, 733
nav_t, 764
obs_t, 788
pcvs_t, 838
sbs_t, 936
solbuf_t, 973
solstatbuf_t, 977
tle_t, 1002
tled_t, 1003
n_bytes
ad936x_iio_samples, 344
n_channels
ad936x_iio_custom, 344
ad936x_iio_samples, 344
n_crc_fail
cnav_v27_part_t, 418
n_decoded
cnav_v27_part_t, 418
n_interleaved_iq_samples
ad936x_iio_samples, 344
n_symbols
cnav_v27_part_t, 418
na
nav_t, 764
rtk_t, 905
namax
nav_t, 764
name
opt_t, 791
sta_t, 986
tled_t, 1003
nav
raw_t, 876
rtcm_t, 903
rtksvr_t, 923
nav_data
Tag, 992
nav_data_id
OSNMA_NavData, 803
nav_message
Galileo_HAS_data, 551
Nav_Message_Packet, 760
nav_message_monitor, 758
∼nav_message_monitor, 758
nav_message_monitor.h, 1671, 1672
nav_message_monitor_sptr
core_libs, 87
Nav_Message_Packet, 758
∼Nav_Message_Packet, 759
nav_message, 760
Nav_Message_Packet, 759
operator=, 760
prn, 760
signal, 760
system, 760
tow_at_current_symbol_ms, 760
nav_message_packet.h, 1673
nav_message_source
Galileo_Ephemeris, 538
nav_message_type
Galileo_Ephemeris, 538
Nav_Message_Udp_Sink, 761
nav_message_udp_sink.h, 1674, 1675
nav_t, 761
alm, 762
cbias, 762
dgps, 762
eph, 762
erp, 762
fcb, 762
geph, 762
glo_cpbias, 762
glo_fcn, 763
ion_bdgim, 763
ion_bdgim_valid, 763
ion_cmp, 763
ion_gal, 763
ion_gps, 763
ion_irn, 763
ion_qzs, 763
lam, 763
leaps, 763
lexeph, 763
lexion, 763
n, 764

na, 764
namax, 764
nc, 764
ncmax, 764
ne, 764
nemax, 764
nf, 764
nfmax, 764
ng, 764
ngmax, 764
nmax, 764
ns, 765
nsmax, 765
nt, 765
ntmax, 765
pclk, 765
pcvs, 765
peph, 765
pppcorr, 765
rbias, 765
sbsion, 765
sbssat, 765
seph, 765
ssr, 766
tec, 766
utc_cmp, 766
utc_gal, 766
utc_glo, 766
utc_gps, 766
utc_irn, 766
utc_qzs, 766
utc_sbs, 766
wlbias, 766
nav_type
Beidou_Cnav1_Ephemeris, 372
Beidou_Dnav_Ephemeris, 387
navsel
rtksvr_t, 923
navsys
prcopt_t, 848
nb
ntrip_t, 785
rtksvr_t, 923
solbuf_t, 973
nb_ar
rtk_t, 905
nb_dk
DSM_KROOT_message, 467
nb_dp
DSM_PKR_message, 469
NB_RTK
algorithms_libs_rtklib, 273
nbit
rtcm_t, 903
nbyte
raw_t, 876
rtcm_t, 903
nc
nav_t, 764
nchannels_in
Obs_Conf, 788
nchannels_out
Obs_Conf, 788
ncmax
nav_t, 764
ndata
tec_t, 997
nddot
tled_t, 1003
ndot
eph_t, 474
tled_t, 1003
ne
nav_t, 764
neb
rtk_t, 905
nemax
nav_t, 764
new_metrics
v27_t, 1028
NEXOBS
algorithms_libs_rtklib, 288
next
half_cyc_tag, 697
kernel_info_t, 724
nf
nav_t, 764
prcopt_t, 848
NF_RTK
algorithms_libs_rtklib, 273
nfix
rtk_t, 905
nfmax
nav_t, 764
NFREQ
algorithms_libs_rtklib, 288
NFREQGLO
algorithms_libs_rtklib, 288
ng
nav_t, 764
ngmax
nav_t, 764
NI_RTK
algorithms_libs_rtklib, 273
nigp
sbsion_t, 939
niter
prcopt_t, 848
NL_RTK
algorithms_libs_rtklib, 273
nmas
DSM_nma_header, 469
MACK_message, 740
Tag, 992
NMAX
rtklib_preceph.h, 1230

nmax
erp_t, 475
lex_t, 733
nav_t, 764
obs_t, 789
pcvs_t, 838
sbs_t, 936
solbuf_t, 973
solstatbuf_t, 977
tle_t, 1002
nmea_dump_devname
Pvt_Conf, 858
nmea_dump_filename
Pvt_Conf, 858
nmea_output_file_enabled
Pvt_Conf, 858
nmea_output_file_path
Pvt_Conf, 859
Nmea_Printer, 767
~Nmea_Printer, 767
get_GPGGA, 767
get_GPGSA, 767
Nmea_Printer, 767
Print_Nmea_Line, 768
nmea_printer.h, 1320
nmea_rate_ms
Pvt_Conf, 859
nmeacycle
rtksvr_t, 923
nmeaintv
solopt_t, 974
nmeapos
rtksvr_t, 923
nmeareq
rtksvr_t, 924
nmsg
rtksvr_t, 924
nmsg2
rtcm_t, 903
nmsg3
rtcm_t, 903
nonlinear_tracking.h, 1622, 1623
Notch, 768
notch_cc.h, 1118, 1119
notch_filter.h, 1113
notch_filter_lite.h, 1114
notch_lite_cc.h, 1120, 1121
notch_lite_sptr
input_filter_gr_blocks, 231
notch_sptr
input_filter_gr_blocks, 231
NotchFilter, 768
connect, 769
disconnect, 769
get_left_block, 769
get_right_block, 769
implementation, 769
item_size, 769
role, 769
NotchFilterLite, 770
connect, 770
disconnect, 770
get_left_block, 770
get_right_block, 771
implementation, 771
item_size, 771
role, 771
NotchLite, 771
NP_PPP
algorithms_libs_rtklib, 273
NP_RTK
algorithms_libs_rtklib, 273
npb
rtksvr_t, 924
npk
DSM_PKR_message, 469
npkt
DSM_PKR_message, 469
Osnma_Public_Key_Material, 805
npkt_valid
Osnma_Public_Key_Material, 805
npktid
DSM_PKR_message, 469
NR_PPP
algorithms_libs_rtklib, 274
NR_RTK
algorithms_libs_rtklib, 274
ns
nav_t, 765
pppcorr_t, 842
sol_t, 971
nsat
msm_h_t, 754
sbssat_t, 941
NSATBDS
algorithms_libs_rtklib, 289
NSATGAL
algorithms_libs_rtklib, 289
NSATGLO
algorithms_libs_rtklib, 289
NSATGPS
algorithms_libs_rtklib, 289
NSATIRN
algorithms_libs_rtklib, 289
NSATLEO
algorithms_libs_rtklib, 289
NSATQZS
algorithms_libs_rtklib, 289
NSATSBS
algorithms_libs_rtklib, 289
nsb
rtksvr_t, 924
nsbs
rtksvr_t, 924
nsig
msm_h_t, 754

nsmax
nav_t, 765
pppcorr_t, 842
nsol
rtksvr_t, 924
nsr_file_signal_source.h, 1437
NsrFileSignalSource, 772
create_file_source_hook, 774
itemTypeToSize, 774
packetsPerSample, 774
pre_connect_hook, 774
pre_disconnect_hook, 774
source, 774
nsta
pppcorr_t, 843
nsvfix
sdr_gnss_packet_t, 942
NSYS
algorithms_libs_rtklib, 289
Nsys
Galileo_HAS_data, 551
Nsys_sub
Galileo_HAS_data, 551
NSYSBDS
algorithms_libs_rtklib, 289
NSYSGAL
algorithms_libs_rtklib, 290
NSYSGLO
algorithms_libs_rtklib, 290
NSYSGPS
algorithms_libs_rtklib, 290
NSYSIRN
algorithms_libs_rtklib, 290
NSYSLEO
algorithms_libs_rtklib, 290
NSYSQZS
algorithms_libs_rtklib, 290
nt
Mack_lookup, 739
nav_t, 765
pppcorr_t, 843
NT_RTK
algorithms_libs_rtklib, 274
ntlab_file_signal_source.h, 1438, 1439
NTLabFileSignalSource, 774
create_file_source_hook, 777
get_left_block, 777
get_right_block, 777
itemTypeToSize, 777
packetsPerSample, 777
pre_connect_hook, 777
pre_disconnect_hook, 777
source, 777
ntmax
nav_t, 765
pppcorr_t, 843
NTRIP_AGENT
algorithms_libs_rtklib, 274
rtklib_stream.h, 1266
ntrip_caster_address
Pvt_Conf, 859
NTRIP_CLI_PORT
algorithms_libs_rtklib, 274
rtklib_stream.h, 1266
ntrip_client_enabled
Pvt_Conf, 859
ntrip_fallback_to_single
Pvt_Conf, 859
ntrip_gga_period_ms
Pvt_Conf, 859
ntrip_max_correction_age_s
Pvt_Conf, 859
NTRIP_MAXRSP
algorithms_libs_rtklib, 274
rtklib_stream.h, 1266
NTRIP_MAXSTR
algorithms_libs_rtklib, 274
rtklib_stream.h, 1266
NTRIP_MIN_COMMON_SATELLITES
Rtklib_Solver, 920
ntrip_mountpoint
Pvt_Conf, 859
ntrip_password
Pvt_Conf, 859
ntrip_password_env
Pvt_Conf, 859
ntrip_port
Pvt_Conf, 859
ntrip_reconnect_interval_ms
Pvt_Conf, 859
NTRIP_RSP_ERROR
algorithms_libs_rtklib, 274
rtklib_stream.h, 1266
NTRIP_RSP_HTTP
algorithms_libs_rtklib, 274
rtklib_stream.h, 1267
NTRIP_RSP_OK_CLI
algorithms_libs_rtklib, 274
rtklib_stream.h, 1267
NTRIP_RSP_OK_SVR
algorithms_libs_rtklib, 275
rtklib_stream.h, 1267
NTRIP_RSP_SRCTBL
algorithms_libs_rtklib, 275
rtklib_stream.h, 1267
NTRIP_RSP_TBLEND
algorithms_libs_rtklib, 275
rtklib_stream.h, 1267
Ntrip_Rtcm_Client, 777
Ntrip_Rtcm_Client_Test_Access, 778
ntrip_rtcm_client.h, 1321, 1322
Ntrip_Rtcm_Client_Config, 778
enabled, 779
gga_period_ms, 779
host, 779
max_age_s, 779

mountpoint, 779
password, 779
port, 779
reconnect_interval_ms, 779
send_gga, 780
station_id, 780
timeout_ms, 780
tls_enabled, 780
username, 780
validate, 779
version, 780
Ntrip_Rtcm_Client_State
pvt_libs, 300
Ntrip_Rtcm_Client_Status, 780
bytes_received, 781
connected, 781
decoder_errors, 781
message, 781
observation_epochs_decoded, 781
reconnect_count, 781
rtcm_messages_decoded, 781
running, 781
state, 781
Ntrip_Rtcm_Client_Test_Access
Ntrip_Rtcm_Client, 778
Ntrip_Rtcm_Snapshot, 781
age_s, 782
antenna_height_m, 782
base_position_ecef_m, 782
base_position_message_type, 782
ephemerides, 782
fresh, 782
generation, 783
has_base_position, 783
has_observations, 783
itrf, 783
observation_time, 783
observations, 783
refresh_age, 782
station_id, 783
ntrip_rtk_is_entry_band
pvt_libs, 300
ntrip_rtk_navigation_systems
pvt_libs, 300
ntrip_rtk_required_bands
pvt_libs, 300
ntrip_rtk_second_band_slot
pvt_libs, 300
Ntrip_Rtk_Signal, 783
band_slot, 784
flag, 784
signal, 784
system, 784
ntrip_rtk_signals
pvt_libs, 301
ntrip_rtk_signals.h, 1325, 1326
ntrip_rtk_supported_sets_description
pvt_libs, 301
ntrip_rtk_supported_signal_mask
pvt_libs, 301
ntrip_rtk_system_label
pvt_libs, 301
ntrip_rtk_valid_channel_set
pvt_libs, 301
ntrip_send_gga
Pvt_Conf, 860
ntrip_station_id
Pvt_Conf, 860
NTRIP_SVR_PORT
algorithms_libs_rtklib, 275
rtklib_stream.h, 1267
ntrip_t, 784
buff, 785
chunk_remaining, 785
chunk_state, 785
chunked, 785
host, 785
mntpnt, 785
nb, 785
passwd, 785
port, 785
request_buff, 786
request_length, 786
request_offset, 786
request_sent, 786
response_received, 786
state, 786
str, 786
tcp, 786
tls_ctx, 786
tls_enabled, 786
type, 786
url, 786
user, 787
v1_retry_requested, 787
version, 787
ntrip_text_has_space_or_control
pvt_libs, 301
ntrip_timeout_ms
Pvt_Conf, 860
ntrip_tls_enabled
Pvt_Conf, 860
ntrip_username
Pvt_Conf, 860
ntrip_version
Pvt_Conf, 860
NTRIP_VERSION_1
rtklib_stream.h, 1267
NTRIP_VERSION_2
rtklib_stream.h, 1267
num_checks
BeidouLdpcGraph, 398
num_codes
Acq_Conf, 334
num_doppler_bins_step2
Acq_Conf, 334

Acq_Conf_Fpga, 338
num_kernels
cl_fft_plan, 413
num_local_mem_banks
cl_fft_plan, 413
NUM_RX_VARIABLES
tracking_libs, 319
NUM_TX_VARIABLES_GALILEO_E1
tracking_libs, 319
NUM_TX_VARIABLES_GPS_L1_CA
tracking_libs, 319
num_variables
BeidouLdpcGraph, 398
num_workgroups
kernel_info_t, 724
num_workitems_per_workgroup
kernel_info_t, 724
num_xforms_per_workgroup
kernel_info_t, 724
NX
rtklib_pntpos.h, 1221
nx
rtk_t, 905
NX_PPP
algorithms_libs_rtklib, 275
NX_RTK
algorithms_libs_rtklib, 275

OB
ion_gsms_stream_encodings.h, 1511
OBA
ion_gsms_stream_encodings.h, 1511
obs
raw_t, 876
rtcm_t, 903
rtksvr_t, 924
obs_adapters, 302
Obs_Conf, 787
always_output_gs, 787
dump, 787
dump_filename, 787
dump_mat, 787
enable_carrier_smoothing, 788
enable_E6, 788
enable_monitor, 788
nchannels_in, 788
nchannels_out, 788
observable_interval_ms, 788
smoothing_factor, 788
obs_conf.h, 1292
obs_gr_blocks, 302
hybrid_observables_gs_sptr, 303
obs_t, 788
data, 788
n, 788
nmax, 789
obsd_t, 789
code, 789
D, 789
L, 789
LLI, 789
Lstd, 789
P, 789
Pstd, 790
rcv, 790
sat, 790
SNR, 790
time, 790
observable_interval_ms
Obs_Conf, 788
Pvt_Conf, 860
Observables, 302
observables_interface.h, 1641, 1642
observables_libs, 303
ObservablesInterface, 790
observation_epochs_decoded
Ntrip_Rtcm_Client_Status, 781
observation_time
Ntrip_Rtcm_Snapshot, 783
observations
Ntrip_Rtcm_Snapshot, 783
obsflag
rtcm_t, 903
obuf
raw_t, 876
odisp
prcopt_t, 849
off
pcv_t, 837
raw_t, 876
offset
file_t, 485
SensorDataConfiguration, 946
OG
ion_gsms_stream_encodings.h, 1512
OGA
ion_gsms_stream_encodings.h, 1512
old_metrics
v27_t, 1028
omega
Gnss_Almanac, 605
Gnss_Ephemeris, 616
Omega0_10
Galileo_Almanac_Helper, 526
OMEGA0_10_LSB
core_system_parameters, 211
Omega0_7
Galileo_Almanac_Helper, 527
OMEGA0_7_LSB
core_system_parameters, 211
Omega0_8
Galileo_Almanac_Helper, 527
OMEGA0_8_LSB
core_system_parameters, 211
omega0_rad
Bds3_B1c_AlmanacMedium, 359
Bds3_B1c_AlmanacReduced, 361

Omega0red
Galileo_Reduced_CED, 566
OMEGA_0
Gnss_Almanac, 605
Gnss_Ephemeris, 616
OMEGA_0_2_LSB
core_system_parameters, 211
OMEGA_0_LSB
core_system_parameters, 211
OMEGA_2_LSB
core_system_parameters, 211
omega_7
Galileo_Almanac_Helper, 527
OMEGA_7_LSB
core_system_parameters, 212
omega_8
Galileo_Almanac_Helper, 527
OMEGA_8_LSB
core_system_parameters, 212
omega_9
Galileo_Almanac_Helper, 527
OMEGA_9_LSB
core_system_parameters, 212
Omega_dot_10
Galileo_Almanac_Helper, 527
OMEGA_DOT_10_LSB
core_system_parameters, 212
OMEGA_DOT_3_LSB
core_system_parameters, 212
Omega_dot_7
Galileo_Almanac_Helper, 527
OMEGA_DOT_7_LSB
core_system_parameters, 212
Omega_dot_8
Galileo_Almanac_Helper, 527
OMEGA_DOT_8_LSB
core_system_parameters, 212
OMEGA_DOT_LSB
core_system_parameters, 212
omega_dot_rad_s
Bds3_B1c_AlmanacMedium, 359
OMEGA_LSB
core_system_parameters, 212
omega_rad
Bds3_B1c_AlmanacMedium, 360
OMEGAdot
Gnss_Almanac, 605
Gnss_Ephemeris, 616
OMG
tled_t, 1003
omg
alm_t, 353
eph_t, 474
tled_t, 1004
OMG0
alm_t, 353
eph_t, 474
OMGd
alm_t, 353
eph_t, 474
only_beidou
Signal_Enabled_Flags, 962
only_galileo
Signal_Enabled_Flags, 962
only_glonass
Signal_Enabled_Flags, 962
only_gps
Signal_Enabled_Flags, 962
only_qzss
Signal_Enabled_Flags, 962
open
SbasL1DumpSession, 935
open_channel
Fpga_Multicorrelator_8sc, 514
open_device
Fpga_Acquisition, 508
opencl_ready
pcps_opencl_acquisition_cc, 822
openpath
file_t, 485
operator<<
Gnss_Satellite, 620
Gnss_Signal, 631
operator()
Allocator, 351
Deleter, 439
std::hash< SensorIdentifier::value_type >, 701
operator+
GPU_Complex, 691
GPU_Complex_Short, 692
operator+=
GPU_Complex, 691
operator=
Exponential_Smoother, 479
Gnss_Satellite, 620
Gnss_Synchro, 633
Nav_Message_Packet, 760
Secure_String, 944
Serdes_Galileo_Eph, 953
Serdes_Gnss_Synchro, 954
Serdes_Gps_Eph, 956
Serdes_Monitor_Pvt, 957, 958
Serdes_Nav_Message, 959
Tracking_loop_filter, 1011
operator==
Gnss_Satellite, 620
Gnss_Signal, 631
operator*
GPU_Complex, 691
GPU_Complex_Short, 692
opt
raw_t, 876
rtcm_t, 903
rtk_t, 906
opt_t, 791
comment, 791

format, 791
name, 791
var, 791
or_all
Signal_Enabled_Flags, 961
orbit_correction_flag
mt1_header, 755
osmosdr_signal_source.h, 1440
OsmosdrSignalSource, 792
connect, 793
disconnect, 793
get_left_block, 793
get_right_block, 793
item_size, 793
OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS
osnma_data.h, 2044
OSNMA_AUTH_IOD_MAX_AGE_SECONDS
osnma_data.h, 2044
osnma_crypto_material.h, 1676
Osnma_Crypto_Material_Manager, 793
OSNMA_data, 794
d_dsm_header, 794
d_dsm_kroot_message, 794
d_dsm_kroot_new_message, 794
d_dsm_pkr_message, 794
d_mack_message, 794
d_nav_data, 795
d_nma_header, 795
osnma_data.h, 2043, 2044
GALILEO_SECONDS_PER_WEEK, 2044
OSNMA_AUTH_IOD_FUTURE_TOLERANCE_SECONDS, 2044
OSNMA_AUTH_IOD_MAX_AGE_SECONDS, 2044
OSNMA_DSM_Reader, 795
osnma_dsm_reader.h, 2048
Osnma_Helper, 796
GST_START_EPOCH, 796
osnma_helper.h, 1677, 1678
Osnma_Merkle_Tree_Material, 796
applicability, 797
hash_function, 797
public_keys, 797
root, 797
source, 797
state, 797
uid, 797
valid, 797
xml_path, 797
Osnma_Merkle_Tree_Material::PublicKeyEntry, 851
key_type, 852
leaf_index, 852
length_bits, 852
pkid, 852
pkid_valid, 852
point, 852
OSNMA_msg, 797
hkroot, 798
mack, 798
page_valid, 798
page_validity_available, 798
PRN, 798
TOW_sf0, 798
WN_sf0, 798
osnma_msg_receiver, 798
~osnma_msg_receiver, 799
msg_handler_osnma, 799
OsnmaTestVectors, 800
read_merkle_xml, 799
set_merkle_root, 799
verify_dsm_pkr, 800
osnma_msg_receiver.h, 1679, 1680
FRIEND_TEST, 1680
osnma_msg_receiver_sptr
core_libs, 87
osnma_nav_data_manager.h, 1685
OSNMA_NavData, 800
clear_tag_accumulation_metadata, 801
continue_tag_accumulation, 801
get_accumulated_tag_adkd, 801
get_first_accumulated_tag_gst, 801
get_IOD_nav, 801
get_last_accumulated_tag_gst, 801
get_last_received_TOW, 801
get_last_received_WN, 801
get_prn_d, 801
get_tow_sf0, 802
get_verified_bits, 802
get_verified_status, 802
get_wn_sf0, 802
has_tag_accumulation, 802
nav_data_id, 803
OSNMA_NavData, 801
reset_tag_accumulation, 802
set_ephemeris_data, 802
set_IOD_nav, 802
set_last_received_TOW, 802
set_last_received_WN, 802
set_prn_d, 802
set_tow_sf0, 803
set_update_verified_bits, 803
set_utc_data, 803
set_verified_status, 803
set_wn_sf0, 803
start_tag_accumulation, 803
OSNMA_NavDataManager, 803
Osnma_Public_Key_Material, 804
applicability, 804
compressed_key, 804
fingerprint_sha256, 804
key_type, 804
npkt, 805
npkt_valid, 805
pem_path, 805
pkid, 805
pkid_valid, 805

product_uid, 805
source, 805
state, 805
valid, 805
osnma_strict
Pvt_Conf, 860
OSNMA_TABLE_1
core_system_parameters, 212
OSNMA_TABLE_10
core_system_parameters, 212
OSNMA_TABLE_11
core_system_parameters, 213
OSNMA_TABLE_15
core_system_parameters, 213
OSNMA_TABLE_16
core_system_parameters, 213
OSNMA_TABLE_2
core_system_parameters, 213
OSNMA_TABLE_3
core_system_parameters, 214
OSNMA_TABLE_5
core_system_parameters, 214
OSNMA_TABLE_6
core_system_parameters, 214
OSNMA_TABLE_7
core_system_parameters, 214
OSNMA_TABLE_8
core_system_parameters, 215
OsnmaTestVectors
osnma_msg_receiver, 800
outb
stream_t, 989
outbt
stream_t, 989
outc
solstat_t, 976
ssat_t, 982
outhead
solopt_t, 975
outopt
solopt_t, 975
output_enabled
Pvt_Conf, 860
output_path
Pvt_Conf, 860
output_rate_ms
Pvt_Conf, 860
outr
stream_t, 989
outsingle
prcopt_t, 849
outtype
raw_t, 876
rtcm_t, 903
overflow_reg
PpsSamplestamp, 844

P
obsd_t, 789
rtk_t, 906
p_dk
DSM_KROOT_message, 467
p_dp
DSM_PKR_message, 470
Pa
rtk_t, 906
packetsPerSample
FileSourceBase, 494
FourBitCpxFileSignalSource, 506
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
TwoBitCpxFileSignalSource, 1016
TwoBitPackedFileSignalSource, 1019
page
raw_t, 877
page_data
Beidou_Cnav1_PageData_Message, 375
page_id
Bds3_B1c_PageCommon, 363
page_valid
OSNMA_msg, 798
page_validity_available
OSNMA_msg, 798
parse_agnss_ref_location
core_system_parameters, 134
parse_agnss_ref_utc_time
core_system_parameters, 134
ParseError
INIReader, 712
part1
cnav_msg_decoder_t, 415
part2
cnav_msg_decoder_t, 415
Pass_Through, 805
connect, 806
disconnect, 806
get_left_block, 806
get_right_block, 806
implementation, 806
item_size, 807
item_type, 807
role, 807
pass_through.h, 1180
passwd
ftp_t, 520
ntrip_t, 785
password
Ntrip_Rtcm_Client_Config, 779
path
file_t, 485
stream_t, 989
url_t, 1026
pbias
ssr_t, 984
pbuf
rtksvr_t, 924
pbuff

raw_t, 877
pbyte
raw_t, 877
pclk
nav_t, 765
pclk_t, 807
clk, 807
index, 807
std, 807
time, 807
pcps_acquisition, 808
cuda_grid_count, 809
cuda_ready, 809
general_work, 809
mag, 809
set_active, 809
set_channel, 809
set_channel_fsm, 810
set_doppler_center, 810
set_doppler_num_bins, 810
set_gnss_synchro, 810
set_local_code, 810
pcps_acquisition.h, 1046, 1047
pcps_acquisition_adapter.h, 1033
pcps_acquisition_adapter_custom.h, 1035
pcps_acquisition_adapter_fpga.h, 1037
pcps_acquisition_fine_doppler_cc, 811
~pcps_acquisition_fine_doppler_cc, 812
general_work, 812
mag, 812
set_active, 812
set_channel, 812
set_channel_fsm, 812
set_gnss_synchro, 813
set_local_code, 813
pcps_acquisition_fine_doppler_cc.h, 1051, 1052
pcps_acquisition_fine_doppler_cc_sptr
acquisition_gr_blocks, 81
pcps_acquisition_fpga, 813
~pcps_acquisition_fpga, 814
init, 814
mag, 814
reset_acquisition, 814
set_active, 814
set_channel, 815
set_channel_fsm, 815
set_doppler_center, 815
set_gnss_synchro, 815
set_local_code, 815
stop_acquisition, 815
pcps_acquisition_fpga.h, 1054, 1055
pcps_acquisition_fpga_sptr
acquisition_gr_blocks, 81
pcps_acquisition_sptr
acquisition_gr_blocks, 81
pcps_assisted_acquisition_cc, 816
~pcps_assisted_acquisition_cc, 817
general_work, 817
mag, 817
set_active, 817
set_channel, 817
set_channel_fsm, 817
set_gnss_synchro, 817
set_local_code, 818
pcps_assisted_acquisition_cc.h, 1057, 1058
pcps_assisted_acquisition_cc_sptr
acquisition_gr_blocks, 81
pcps_cccwsr_acquisition_cc, 818
~pcps_cccwsr_acquisition_cc, 819
general_work, 819
mag, 819
set_active, 819
set_channel, 820
set_channel_fsm, 820
set_gnss_synchro, 820
set_local_code, 820
pcps_cccwsr_acquisition_cc.h, 1060, 1061
pcps_cccwsr_acquisition_cc_sptr
acquisition_gr_blocks, 82
pcps_opencl_acquisition_cc, 821
~pcps_opencl_acquisition_cc, 822
general_work, 822
mag, 822
opencl_ready, 822
set_active, 822
set_channel, 822
set_channel_fsm, 822
set_gnss_synchro, 823
set_local_code, 823
pcps_opencl_acquisition_cc.h, 1063, 1065
CL_SILENCE_DEPRECATION, 1064
pcps_opencl_acquisition_cc_sptr
acquisition_gr_blocks, 82
pcps_quicksync_acquisition_cc, 823
~pcps_quicksync_acquisition_cc, 824
general_work, 824
mag, 824
set_active, 824
set_channel, 825
set_channel_fsm, 825
set_gnss_synchro, 825
set_local_code, 825
pcps_quicksync_acquisition_cc.h, 1067, 1068
pcps_quicksync_acquisition_cc_sptr
acquisition_gr_blocks, 82
pcps_tong_acquisition_cc, 826
~pcps_tong_acquisition_cc, 827
general_work, 827
mag, 827
set_active, 827
set_channel, 827
set_channel_fsm, 827
set_gnss_synchro, 827
set_local_code, 828
pcps_tong_acquisition_cc.h, 1071, 1072
pcps_tong_acquisition_cc_sptr

acquisition_gr_blocks, 82
PcpsAcquisitionAdapter, 828
connect, 829
disconnect, 829
get_left_block, 829
get_right_block, 829
implementation, 829
item_size, 830
mag, 830
reset, 830
role, 830
set_channel, 830
set_channel_fsm, 830
set_doppler_center, 830
set_doppler_num_bins, 830
set_gnss_synchro, 830
set_local_code, 831
set_resampler_latency, 831
stop_acquisition, 831
PcpsAcquisitionAdapterCustom, 831
connect, 832
disconnect, 832
get_left_block, 832
get_right_block, 832
implementation, 833
item_size, 833
mag, 833
reset, 833
role, 833
set_channel, 833
set_channel_fsm, 833
set_gnss_synchro, 833
set_local_code, 833
set_resampler_latency, 834
stop_acquisition, 834
PcpsAcquisitionAdapterFpga, 834
connect, 835
disconnect, 835
get_left_block, 835
get_right_block, 835
implementation, 835
item_size, 835
mag, 835
reset, 836
role, 836
set_channel, 836
set_channel_fsm, 836
set_doppler_center, 836
set_gnss_synchro, 836
set_local_code, 836
set_resampler_latency, 836
stop_acquisition, 836
pcv
pcvs_t, 838
pcv_t, 837
code, 837
off, 837
sat, 837
te, 837
ts, 837
type, 837
var, 837
pcvr
prcopt_t, 849
pcvs
nav_t, 765
pcvs_t, 838
n, 838
nmax, 838
pcv, 838
pdop
Monitor_Pvt, 751
pem_path
Osnma_Public_Key_Material, 805
pending_messages
Sbas_Rtklib_Corrections, 935
peph
nav_t, 765
peph_t, 838
cov, 838
index, 838
pos, 838
std, 839
time, 839
vco, 839
vel, 839
vst, 839
perr
exterr_t, 480
pfa
Acq_Conf, 335
pfa2
Acq_Conf, 335
ph
ssat_t, 982
phase_bias
Galileo_HAS_data, 552
phase_bias_cycle
HAS_obs_corrections, 699
phase_bias_discontinuity
HAS_obs_corrections, 699
phase_bias_flag
mt1_header, 755
phase_discontinuity_indicator
Galileo_HAS_data, 552
HAS_obs_corrections, 699
phase_unwrap
tracking_libs, 322
phi0_rad
Bds3_B1c_AlmanacReduced, 361
phw
ssat_t, 982
PI_TWO_N19
core_system_parameters, 215
PI_TWO_N23
core_system_parameters, 215

PI_TWO_N31
core_system_parameters, 215
PI_TWO_N38
core_system_parameters, 215
PI_TWO_N43
core_system_parameters, 215
pkid
DSM_KROOT_message, 467
Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
Osnma_Public_Key_Material, 805
pkid_valid
Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
Osnma_Public_Key_Material, 805
plen
raw_t, 877
pll_bw_hz
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 455
pll_bw_narrow_hz
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
pll_cloop_two_quadrant_atan
tracking_libs, 322
pll_filter_order
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
pll_four_quadrant_atan
tracking_libs, 322
pll_pull_in_bw_hz
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
plutosdr_signal_source.h, 1441, 1442
PlutosdrSignalSource, 839
connect, 840
disconnect, 840
get_left_block, 840
get_right_block, 840
item_size, 840
pm_x
Gps_CNAV_Eop, 651
pm_x_arcsec
Bds3_B1c_Eop, 362
pm_x_dot
Gps_CNAV_Eop, 651
pm_x_dot_arcsec_day
Bds3_B1c_Eop, 362
pm_y
Gps_CNAV_Eop, 651
pm_y_arcsec
Bds3_B1c_Eop, 362
pm_y_dot
Gps_CNAV_Eop, 651
pm_y_dot_arcsec_day
Bds3_B1c_Eop, 362
PMODE_DGPS
algorithms_libs_rtklib, 290
PMODE_FIXED
algorithms_libs_rtklib, 290
PMODE_KINEMA
algorithms_libs_rtklib, 290
PMODE_MOVEB
algorithms_libs_rtklib, 290
PMODE_PPP_FIXED
algorithms_libs_rtklib, 290
PMODE_PPP_KINEMA
algorithms_libs_rtklib, 291
PMODE_PPP_STATIC
algorithms_libs_rtklib, 291
PMODE_SINGLE
algorithms_libs_rtklib, 291
PMODE_STATIC
algorithms_libs_rtklib, 291
pntpos
rtklib_pntpos.h, 1220
Pnum2
Beidou_Dnav_Differential_Corrections, 382
pocket_sdr_signal_source.h, 1443, 1444
PocketSdrSignalSource, 841
connect, 842
disconnect, 842
get_left_block, 842
get_right_block, 842
item_size, 842
point
Osnma_Merkle_Tree_Material::PublicKeyEntry, 852
poly
v27_t, 1028
POLYCRC24Q
algorithms_libs_rtklib, 291
POLYCRC32
algorithms_libs_rtklib, 291
pop_front
algorithms_libs, 251
port
Ntrip_Rtcm_Client_Config, 779
ntrip_t, 785
stream_t, 989
tcp_t, 994
pos
geph_t, 579
lexeph_t, 734
peph_t, 838
seph_t, 951
sta_t, 986
pos0
lexion_t, 735
pos_x
Monitor_Pvt, 751
pos_y
Monitor_Pvt, 751
pos_z
Monitor_Pvt, 751
posf

solopt_t, 975
posopt
prcopt_t, 849
POSOPT_RINEX
algorithms_libs_rtklib, 291
post_disconnect_hook
SpirFileSignalSource, 980
pppcorr
nav_t, 765
pppcorr_t, 842
ns, 842
nsmax, 842
nsta, 843
nt, 843
ntmax, 843
rr, 843
stas, 843
stec, 843
trop, 843
pppopt
prcopt_t, 849
pps_samplestamp.h, 1515
pps_tcp_rx, 843
is_connected, 844
PpsSamplestamp, 844
overflow_reg, 844
samplestamp, 844
ppstcprx.h, 1516
prc
dgps_t, 439
sbsfcorr_t, 937
prCA
raw_t, 877
PRCOPT_DEFAULT
rtklib_rtksvr.h, 1253
prcopt_t, 844
antdel, 845
anttype, 845
arfilter, 846
armaxiter, 846
armaxposvar, 846
bancroft_init, 846
baseline, 846
bdsmodear, 846
codesmooth, 846
dynamics, 846
elmaskar, 846
elmaskhold, 846
elmin, 846
eratio, 846
err, 847
estqzsisb, 847
exsats, 847
exterr, 847
freqopt, 847
glomodear, 847
initrst, 847
intpref, 847
ionoopt, 847
maxaveep, 847
maxgdop, 847
maxinno, 847
maxout, 848
maxtdiff, 848
mindropsats, 848
minfix, 848
minfixsats, 848
minholdsats, 848
minlock, 848
mode, 848
modear, 848
navsys, 848
nf, 848
niter, 848
odisp, 849
outsingle, 849
pcvr, 849
posopt, 849
pppopt, 849
prn, 849
rb, 849
refpos, 849
rnxopt, 849
rovpos, 849
ru, 849
sateph, 849
sbascorr, 850
sbassatsel, 850
sclkstab, 850
snrmask, 850
soltype, 850
std, 850
syncsol, 850
thresar, 850
thresdop, 850
thresslip, 850
tidecorr, 850
tropopt, 850
varholdamb, 851
prcout
rtksvr_t, 924
pre_connect_hook
FileTimestampSignalSource, 497
FourBitCpxFileSignalSource, 506
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
SpirFileSignalSource, 980
TwoBitCpxFileSignalSource, 1017
TwoBitPackedFileSignalSource, 1019
pre_disconnect_hook
FileTimestampSignalSource, 497
FourBitCpxFileSignalSource, 507
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
TwoBitCpxFileSignalSource, 1017
TwoBitPackedFileSignalSource, 1019

preamble_detection_statistic
Beidou_Cnav2_Navigation_Message, 378
preamble_seen
cnav_v27_part_t, 418
predicted_doppler
Gnss_Almanac, 604
Gnss_Ephemeris, 612
prev_ratio1
sol_t, 972
prev_ratio2
sol_t, 972
print_IGM01
Rtcm, 890
print_IGM02
Rtcm, 890
print_IGM03
Rtcm, 890
print_IGM05
Rtcm, 890
print_MSM_1
Rtcm, 890
print_MSM_2
Rtcm, 891
print_MSM_3
Rtcm, 891
print_MSM_4
Rtcm, 891
print_MSM_5
Rtcm, 892
print_MSM_6
Rtcm, 892
print_MSM_7
Rtcm, 892
print_MT1001
Rtcm, 893
print_MT1002
Rtcm, 893
print_MT1003
Rtcm, 893
print_MT1004
Rtcm, 893
print_MT1005
Rtcm, 893
print_MT1005_test
Rtcm, 894
Rtcm_Printer, 901
print_MT1006
Rtcm, 894
print_MT1008
Rtcm, 894
print_MT1009
Rtcm, 894
print_MT1010
Rtcm, 895
print_MT1011
Rtcm, 895
print_MT1012
Rtcm, 896
print_MT1019
Rtcm, 896
print_MT1020
Rtcm, 896
print_MT1029
Rtcm, 897
print_MT1045
Rtcm, 897
print_MT1057
Rtcm, 897
print_MT1058
Rtcm, 897
print_MT1059
Rtcm, 897
print_MT1060
Rtcm, 898
Print_Nmea_Line
Nmea_Printer, 768
print_packet
An_Packet_Printer, 355
print_rinex_annotation
Rinex_Printer, 884
Print_Rtcm_Messages
Rtcm_Printer, 901
priorize_satellites
GNSSFlowgraph, 643
PRN
Beidou_Cnav1_PageData_Message, 375
Beidou_Dnav_Differential_Correction, 381
Galileo_Reduced_CED, 566
Glonass_Gnav_Almanac, 583
Glonass_Gnav_Ephemeris, 593
Gnss_Almanac, 606
Gnss_Ephemeris, 616
Gnss_Synchro, 636
Gps_Acq_Assist, 647
Gps_CNAV_Eop, 651
OSNMA_msg, 798
prn
Bds3_B1c_AlmanacMedium, 360
Bds3_B1c_AlmanacReduced, 361
cnav_msg_t, 416
lexmsg_t, 735
Monitor_Pvt::TrackedSatelliteInfo, 1008
Nav_Message_Packet, 760
prcopt_t, 849
Sbas_Raw_Message, 934
sbsmsg_t, 940
TOW_to_trk, 1007
PRN_d
MACK_tag_info, 741
Tag, 992
PRN_HWBIA
algorithms_libs_rtklib, 291
PRN_HWBIAS
algorithms_libs_rtklib, 291
PRNa
MACK_message, 740

Tag, 992
proc_pack_carr_error
Tcp_Packet_Data, 993
proc_pack_carrier_doppler_hz
Tcp_Packet_Data, 993
proc_pack_code_error
Tcp_Packet_Data, 993
process_test_page
galileo_e6_has_msg_receiver, 533
processed_control_messages
ControlThread, 428
product_uid
Osnma_Public_Key_Material, 805
prog
solopt_t, 975
program
cl_fft_plan, 413
Prompt_I
Gnss_Synchro, 636
Prompt_Q
Gnss_Synchro, 636
property
FileConfiguration, 486–488
InMemoryConfiguration, 713, 714
proto
ftp_t, 520
Protobuf_Cleanup_Manager, 851
get, 851
protobuf_cleanup_manager.h, 1686
protobuf_enabled
Pvt_Conf, 860
proxy_status
fpga_dma-proxy.h, 1495
PRUNIT_GLO
rtklib_rtcm3.h, 1237
PRUNIT_GPS
rtklib_rtcm3.h, 1237
Pseudorange_m
Gnss_Synchro, 636
Pstd
obsd_t, 790
pt
ssat_t, 982
public_keys
Osnma_Merkle_Tree_Material, 797
pull_in_time_s
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
pulse_blanking_cc, 852
pulse_blanking_cc.h, 1122
pulse_blanking_cc_sptr
input_filter_gr_blocks, 231
pulse_blanking_filter.h, 1115, 1116
PulseBlankingFilter, 853
connect, 853
disconnect, 853
get_left_block, 853
get_right_block, 854
implementation, 854
item_size, 854
role, 854
push
Concurrent_Queue< Data >, 421
push_back
algorithms_libs, 251
pv_Geo_to_ECEF
algorithms_libs, 251
PVT, 259
pvt_adapters, 297
Pvt_Conf, 854
an_dump_devname, 856
an_output_enabled, 856
an_rate_ms, 856
display_rate_ms, 856
dump, 856
dump_filename, 856
dump_mat, 856
enable_pvt_kf, 856
enable_rx_clock_correction, 856
flag_nmea_tty_port, 856
flag_rtcm_server, 857
flag_rtcm_tty_port, 857
geojson_output_enabled, 857
geojson_output_path, 857
geojson_rate_ms, 857
glonass_strict_health, 857
gpx_output_enabled, 857
gpx_output_path, 857
gpx_rate_ms, 857
has_output_file_path, 857
kf_use_imu_vel, 857
kml_output_enabled, 857
kml_output_path, 858
kml_rate_ms, 858
log_source_timetag, 858
log_source_timetag_file, 858
max_obs_block_rx_clock_offset_ms, 858
measures_ecef_pos_sd_m, 858
measures_ecef_vel_sd_ms, 858
monitor_enabled, 858
monitor_ephemeris_enabled, 858
nmea_dump_devname, 858
nmea_dump_filename, 858
nmea_output_file_enabled, 858
nmea_output_file_path, 859
nmea_rate_ms, 859
ntrip_caster_address, 859
ntrip_client_enabled, 859
ntrip_fallback_to_single, 859
ntrip_gga_period_ms, 859
ntrip_max_correction_age_s, 859
ntrip_mountpoint, 859
ntrip_password, 859
ntrip_password_env, 859
ntrip_port, 859

ntrip_reconnect_interval_ms, 859
ntrip_send_gga, 860
ntrip_station_id, 860
ntrip_timeout_ms, 860
ntrip_tls_enabled, 860
ntrip_username, 860
ntrip_version, 860
observable_interval_ms, 860
osnma_strict, 860
output_enabled, 860
output_path, 860
output_rate_ms, 860
protobuf_enabled, 860
ref_gps_week, 861
rinex_name, 861
rinex_output_enabled, 861
rinex_output_path, 861
rinex_version, 861
rinexobs_rate_ms, 861
rtcm_dump_devname, 861
rtcm_msg_rate_ms, 861
rtcm_output_file_enabled, 861
rtcm_output_file_path, 861
rtcm_station_id, 861
rtcm_tcp_port, 861
rtk_trace_level, 862
show_local_time_zone, 862
signal_enabled_flags, 862
system_ecef_pos_sd_m, 862
system_ecef_vel_sd_ms, 862
udp_addresses, 862
udp_eph_addresses, 862
udp_eph_port, 862
udp_ports, 862
use_e6_for_pvt, 862
use_has_corrections, 862
use_unhealthy_sats, 862
xml_output_enabled, 863
xml_output_path, 863
pvt_conf.h, 1329
pvt_gr_blocks, 297
rtklib_pvt_gs_sptr, 297
pvt_interface.h, 1642, 1643
Pvt_Kf, 863
pvt_kf.h, 1331
pvt_libs, 297
b_io_context, 299
make_ntrip_rtcm_client_config, 300
Ntrip_Rtcm_Client_State, 300
ntrip_rtk_is_entry_band, 300
ntrip_rtk_navigation_systems, 300
ntrip_rtk_required_bands, 300
ntrip_rtk_second_band_slot, 300
ntrip_rtk_signals, 301
ntrip_rtk_supported_sets_description, 301
ntrip_rtk_supported_signal_mask, 301
ntrip_rtk_system_label, 301
ntrip_rtk_valid_channel_set, 301
ntrip_text_has_space_or_control, 301
Rtklib_Fixed_Base_Status, 300
secure_clear_ntrip_credentials, 302
pvt_sol
Rtklib_Solver, 920
Pvt_Solution, 863
get_clock_drift_ppm, 865
get_course_over_ground, 865
get_height, 865
get_latitude, 865
get_longitude, 865
get_num_valid_observations, 865
get_ref_gps_week, 865
get_speed_over_ground, 865
get_time_offset_s, 865
set_clock_drift_ppm, 865
set_course_over_ground, 865
set_num_valid_observations, 865
set_ref_gps_week, 866
set_rx_pos, 866
set_rx_vel, 866
set_speed_over_ground, 866
set_time_offset_s, 866
pvt_solution.h, 1332, 1333
pvt_ssat
Rtklib_Solver, 920
PvtInterface, 866
get_beidou_cnav1_ephemeris, 867
get_beidou_cnav2_ephemeris, 867

qmboc
Acq_Conf, 335
Dll_Pll_Conf, 449
qr
sol_t, 972
qzss.h, 2050
QZSS_CA_PREAMBLE_SYMBOLS_STR
core_system_parameters, 215
Qzss_CNAV_Eop, 868
Qzss_CNAV_Eop, 868
qzss_cnav_eop
Rtklib_Solver, 921
qzss_cnav_eop.h, 2051, 2052
Qzss_CNAV_Iono, 869
Qzss_CNAV_Iono, 870
qzss_cnav_iono
Rtklib_Solver, 921
qzss_cnav_iono.h, 2052
Qzss_CNAV_Utc_Model, 870
Qzss_CNAV_Utc_Model, 871
qzss_cnav_utc_model
Rtklib_Solver, 921
qzss_cnav_utc_model.h, 2053
Qzss_Iono, 871
Qzss_Iono, 873
qzss_iono
Rtklib_Solver, 921
qzss_iono.h, 2054
qzss_l1_code_gen_complex_sampled

algorithms_libs, 251
qzss_l1_code_gen_float
algorithms_libs, 252
qzss_l1_code_is_boc
algorithms_libs, 252
qzss_l1cb_prn_to_nominal_prn
core_system_parameters, 135
qzss_l5i_code_gen_complex_sampled
algorithms_libs, 252
qzss_l5i_code_gen_float
algorithms_libs, 252
QZSS_L5I_NH_CODE_STR
core_system_parameters, 215
qzss_l5q_code_gen_complex_sampled
algorithms_libs, 252
qzss_l5q_code_gen_float
algorithms_libs, 252
QZSS_L5Q_NH_CODE_STR
core_system_parameters, 216
qzss_signal_replica.h, 1181, 1182
Qzss_Utc_Model, 873
Qzss_Utc_Model, 874
qzss_utc_model
Rtklib_Solver, 921
qzss_utc_model.h, 2055

r
GPU_Complex, 691
GPU_Complex_Short, 692
R2D
core_system_parameters, 216
radial_m
HAS_orbit_corrections, 700
range_corrections_valid
Beidou_Dnav_Differential_Corrections, 382
RANGE_MS
rtklib_rtcm3.h, 1237
ratio
sol_t, 972
raw
rtksvr_t, 924
raw_array_signal_source.h, 1445
raw_msg
cnav_msg_t, 416
raw_t, 874
buff, 875
dpCA, 875
ephsat, 875
flag, 875
freqn, 875
halfc, 875
icpc, 875
icpp, 875
iod, 876
len, 876
lexmsg, 876
lockt, 876
msgtype, 876
nav, 876
nbyte, 876
obs, 876
obuf, 876
off, 876
opt, 876
outtype, 876
page, 877
pbuff, 877
pbyte, 877
plen, 877
prCA, 877
receive_time, 877
reply, 877
sbsmsg, 877
sta, 877
subfrm, 877
tbase, 877
time, 877
tobs, 878
tod, 878
week, 878
RawArraySignalSource, 878
connect, 879
disconnect, 879
get_left_block, 879
get_right_block, 879
item_size, 879
rb
prcopt_t, 849
rtk_t, 906
solbuf_t, 973
tec_t, 997
rbias
nav_t, 765
rcv
obsd_t, 790
RE_WGS84
algorithms_libs_rtklib, 291
read
Concurrent_Map< Data >, 420
read_acquisition_results
Fpga_Acquisition, 508
read_bits
IONGSMSChunkUnpackingCtx< WT >, 717
read_lsb_first_
IONGSMSChunkUnpackingCtx< WT >, 718
read_merkle_xml
osnma_msg_receiver, 799
read_MT1005
Rtcm, 898
read_MT1019
Rtcm, 898
read_MT1020
Rtcm, 898
read_MT1045
Rtcm, 899
read_reg32
Fpga_spidev, 516

read_sample_counter
Fpga_Multicorrelator_8sc, 515
readProtobuffer
Serdes_Galileo_Eph, 953
Serdes_Gnss_Synchro, 955
Serdes_Gps_Eph, 956
Serdes_Monitor_Pvt, 958
Serdes_Nav_Message, 959
real
cIFFT_Complex, 414
cIFFT_SplitComplex, 415
receive_time
raw_t, 877
received_tag
Tag, 992
reconnect_count
Ntrip_Rtcm_Client_Status, 781
reconnect_interval_ms
Ntrip_Rtcm_Client_Config, 779
recsno
sta_t, 986
rectype
sta_t, 986
recver
sta_t, 986
reduced_almanac
Bds3_B1c_PageData, 364
reed_solomon.h, 2056
ReedSolomon, 880
decode, 881
encode_with_generator_matrix, 881
encode_with_generator_poly, 881
ReedSolomon, 880
ref_gps_week
Pvt_Conf, 861
refd
ssr_t, 984
Reference Documents, 17
reference_bin_min_sidelobes
Acq_Conf, 335
reference_time
Galileo_Reduced_CED, 565
refpos
prcopt_t, 849
refresh_age
Ntrip_Rtcm_Snapshot, 782
Region1_flag
Galileo_Iono, 559
Region2_flag
Galileo_Iono, 559
Region3_flag
Galileo_Iono, 559
Region4_flag
Galileo_Iono, 559
Region5_flag
Galileo_Iono, 560
rejc
solstat_t, 976
ssat_t, 982
REL_HUMI
algorithms_libs_rtklib, 292
remove_dat
Tlm_Conf, 1005
repeat
FileSourceBase, 494
GSS6450FileSignalSource, 695
MultichannelFileSignalSource, 757
SensorDataFileConfiguration, 948
repeat_satellite
Acq_Conf_Fpga, 338
reply
raw_t, 877
repmode
file_t, 485
request
Gnss_Sdr_Supl_Client, 628
request_buff
ntrip_t, 786
request_length
ntrip_t, 786
request_offset
ntrip_t, 786
request_sent
ntrip_t, 786
resample_sensor_data_tags
algorithms_libs, 252
resampled_fs
Acq_Conf, 335
Acq_Conf_Fpga, 338
Resampler, 303
resampler
algorithms_libs, 253
ArraySignalConditioner, 358
SignalConditioner, 965
resampler_adapters, 303
resampler_gr_blocks, 303
direct_resampler_conditioner_cb_sptr, 304
direct_resampler_conditioner_cc_sptr, 304
direct_resampler_conditioner_cs_sptr, 304
resampler_latency_samples
Acq_Conf, 335
resampler_ratio
Acq_Conf, 335
resc
solstat_t, 976
ssat_t, 982
reserved
DSM_KROOT_message, 468
DSM_nma_header, 469
Galileo_HAS_page, 554
mt1_header, 755
sdr_gnss_packet_t, 942
reserved1
DSM_KROOT_message, 468
reset
AcquisitionInterface, 342

algorithms_libs, 253
beidou_b1c_telemetry_decoder_gs, 368
beidou_b2a_telemetry_decoder_gs, 369
beidou_dnav_telemetry_decoder_gs, 395
dll_pll_veml_tracking_fpga, 459
galileo_telemetry_decoder_gs, 568
glonass_gnav_telemetry_decoder_gs, 598
gps_l1_ca_telemetry_decoder_gs, 673
gps_l2c_telemetry_decoder_gs, 675
gps_l5_telemetry_decoder_gs, 676
HistogramBitSynchronizer, 704
HybridObservables, 706
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
Rtklib_Pvt, 911
sbas_l1_telemetry_decoder_gs, 933
telemetry_impl_interface, 999
TelemetryDecoderAdapter, 1000
Viterbi_Decoder, 1029
reset_acquisition
Fpga_Acquisition, 509
pcps_acquisition_fpga, 814
reset_tag_accumulation
OSNMA_NavData, 802
resp
solstat_t, 977
ssat_t, 982
response_received
ntrip_t, 786
rev
tled_t, 1004
rfport
stream_cfg, 988
rinex_name
Pvt_Conf, 861
rinex_output_enabled
Pvt_Conf, 861
rinex_output_path
Pvt_Conf, 861
Rinex_Printer, 881
∼Rinex_Printer, 882
get_navfilename, 882
get_obsfilename, 882
is_rinex_header_written, 882
log_rinex_nav_bds_cnav1, 883
log_rinex_nav_bds_cnav2, 883
log_rinex_nav_bds_dnav, 883
log_rinex_nav_gal_nav, 883
log_rinex_nav_glo_gnav, 883
log_rinex_nav_gps_cnav, 883
log_rinex_nav_gps_nav, 884
print_rinex_annotation, 884
Rinex_Printer, 882
rinex_printer.h, 1334, 1335
rinex_version
Pvt_Conf, 861
rinexobs_rate_ms
Pvt_Conf, 861
rms
tec_t, 997
rnxopt
prcopt_t, 849
role
ArraySignalConditioner, 358
BeamformerFilter, 367
ByteToShort, 403
Channel, 406
CshortToGrComplex, 433
DirectResamplerConditioner, 443
DllPllTrackingAdapter, 462
DllPllTrackingAdapterFpga, 464
FirFilter, 499
FreqXlatingFirFilter, 519
GalileoE1TcpConnectorTracking, 573
GenSignalSource, 576
GNSSBlockInterface, 640
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
HybridObservables, 706
IbyteToCbyte, 708
IbyteToComplex, 709
IbyteToCshort, 711
IshortToComplex, 722
IshortToCshort, 723
MmseResamplerConditioner, 747
NotchFilter, 769
NotchFilterLite, 771
Pass_Through, 807
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
PulseBlankingFilter, 854
Rtklib_Pvt, 911
SignalConditioner, 965
SignalGenerator, 967
SignalSourceBase, 969
TelemetryDecoderAdapter, 1000
root
Osnma_Merkle_Tree_Material, 797
ROUND_LAMBDA
rtklib_lambda.h, 1217
ROUND_PPP
rtklib_ppp.h, 1225
rovpos
prcopt_t, 849
row_weight
BeidouLdpcGraph, 398
rr
pppcorr_t, 843
sol_t, 972
rrc
dgps_t, 439
sbsfcorr_t, 937

Rtcm, 884
bin_to_binary_data, 888
bin_to_double, 888
bin_to_hex, 888
bin_to_uint, 888
binary_data_to_bin, 888
check_CRC, 888
hex_to_bin, 888
hex_to_int, 888
hex_to_uint, 889
is_server_running, 889
lock_time, 889, 890
print_IGM01, 890
print_IGM02, 890
print_IGM03, 890
print_IGM05, 890
print_MSM_1, 890
print_MSM_2, 891
print_MSM_3, 891
print_MSM_4, 891
print_MSM_5, 892
print_MSM_6, 892
print_MSM_7, 892
print_MT1001, 893
print_MT1002, 893
print_MT1003, 893
print_MT1004, 893
print_MT1005, 893
print_MT1005_test, 894
print_MT1006, 894
print_MT1008, 894
print_MT1009, 894
print_MT1010, 895
print_MT1011, 895
print_MT1012, 896
print_MT1019, 896
print_MT1020, 896
print_MT1029, 897
print_MT1045, 897
print_MT1057, 897
print_MT1058, 897
print_MT1059, 897
print_MT1060, 898
read_MT1005, 898
read_MT1019, 898
read_MT1020, 898
read_MT1045, 899
Rtcm, 887
run_server, 899
send_message, 899
stop_server, 899
rtcm
rtksvr_t, 924
rtcm.h, 1339, 1340
RTCM2PREAMB
rtklib_rtcm.h, 1232
RTCM3PREAMB
rtklib_rtcm.h, 1232
rtcm_dump_devname
Pvt_Conf, 861
rtcm_messages_decoded
Ntrip_Rtcm_Client_Status, 781
rtcm_msg_rate_ms
Pvt_Conf, 861
rtcm_output_file_enabled
Pvt_Conf, 861
rtcm_output_file_path
Pvt_Conf, 861
Rtcm_Printer, 899
∼Rtcm_Printer, 900
lock_time, 900
print_MT1005_test, 901
Print_Rtcm_Messages, 901
Rtcm_Printer, 900
rtcm_printer.h, 1361
rtcm_station_id
Pvt_Conf, 861
rtcm_t, 901
buff, 902
cp, 902
dgps, 902
ephsat, 902
len, 902
lltime, 902
lock, 902
loss, 902
msg, 903
msgtype, 903
msmtype, 903
nav, 903
nbit, 903
nbyte, 903
nmsg2, 903
nmsg3, 903
obs, 903
obsflag, 903
opt, 903
outtype, 903
seqno, 904
ssr, 904
sta, 904
stah, 904
staid, 904
time, 904
time_s, 904
word, 904
rtcm_tcp_port
Pvt_Conf, 861
rtk
rtksvr_t, 924
rtk_lock
algorithms_libs_rtklib, 275
rtk_t, 904
ambc, 905
errbuf, 905
excsat, 905

holdamb, 905
na, 905
nb_ar, 905
neb, 905
nfix, 905
nx, 905
opt, 906
P, 906
Pa, 906
rb, 906
sol, 906
ssat, 906
tt, 906
x, 906
xa, 906
rtk_trace_level
Pvt_Conf, 862
rtk_unlock
algorithms_libs_rtklib, 275
RTKLIB, 297
rtklib.h, 1183, 1193
rtklib_conversions.h, 1209, 1210
rtklib_ephemeris.h, 1212, 1213
BDS_EPH_SELECTION_CNAV1_PREFERRED, 1213
Rtklib_Fixed_Base_Status
pvt_libs, 300
rtklib_ionex.h, 1214, 1216
MIN_EL, 1215
MIN_HGT, 1215
VAR_NOTEC, 1215
rtklib_lambda.h, 1217, 1218
LOOPMAX, 1218
ROUND_LAMBDA, 1217
SGN_LAMBDA, 1217
SWAP_LAMBDA, 1218
rtklib_pntpos.h, 1219, 1221
ERR_ION, 1221
ERR_TROP, 1221
MAXITR, 1221
NX, 1221
pntpos, 1220
VARERR_MIN_EL, 1221
rtklib_ppp.h, 1223, 1227
CONST_AMB, 1226
LOG_PI, 1226
MIN_ARC_GAP, 1226
MIN_PPP, 1225
ROUND_PPP, 1225
SQRT2, 1226
SWAP_D, 1225
SWAP_I, 1225
THRES_RES, 1226
VAR_BIAS, 1226
VAR_CLK, 1226
VAR_GRA_PPP, 1226
VAR_IONO_OFF, 1226
VAR_POS_PPP, 1226
VAR_ZTD, 1226
rtklib_preceph.h, 1229, 1230
EXTERR_CLK, 1230
EXTERR_EPH, 1230
MAXDTE, 1230
NMAX, 1230
Rtklib_Pvt, 906
clear_ephemeris, 909
clear_ephemeris_keep_almanac, 909
connect, 909
disconnect, 909
get_beidou_cnav1_ephemeris, 909
get_beidou_cnav2_ephemeris, 909
get_beidou_dnav_almanac, 909
get_beidou_dnav_ephemeris, 909
get_galileo_almanac, 910
get_galileo_ephemeris, 910
get_glonass_almanac, 910
get_glonass_ephemeris, 910
get_glonass_utc_model, 910
get_gps_almanac, 910
get_gps_cnav_ephemeris, 910
get_gps_ephemeris, 910
get_latest_PVT, 910
get_left_block, 910
get_right_block, 911
implementation, 911
item_size, 911
reset, 911
role, 911
rtklib_pvt.h, 1293
rtklib_pvt_gs, 911
~rtklib_pvt_gs, 912
clear_ephemeris, 912
clear_ephemeris_keep_almanac, 912
get_beidou_dnav_almanac_map, 913
get_beidou_dnav_ephemeris_map, 913
get_galileo_almanac_map, 913
get_galileo_ephemeris_map, 913
get_gps_almanac_map, 913
get_gps_cnav_ephemeris_map, 913
get_latest_PVT, 913
interpolate_observable, 913
work, 913
rtklib_pvt_gs.h, 1297, 1298
rtklib_pvt_gs_sptr
pvt_gr_blocks, 297
rtklib_rtcm.h, 1231, 1232
RTCM2PREAMB, 1232
RTCM3PREAMB, 1232
rtklib_rtcm2.h, 1233, 1234
rtklib_rtcm3.h, 1234, 1238
CODES_BDS, 1236
CODES_GAL, 1236
CODES_GLO, 1237
CODES_GPS, 1237
CODES_QZS, 1237
CODES_SBS, 1237

PRUNIT_GLO, 1237
PRUNIT_GPS, 1237
RANGE_MS, 1237
SSRUDINT, 1237
rtklib_rtkcmn.h, 1240, 1245
Rx, 1244
Ry, 1244
Rz, 1245
rtklib_rtkpos.h, 1248, 1250
rtklib_rtksvr.h, 1252, 1254
PRCOPT_DEFAULT, 1253
SOLOPT_DEFAULT, 1254
rtklib_sbas.h, 1255, 1257
IGPBAND1, 1257
IGPBAND2, 1257
WEEKOFFSET, 1257
rtklib_secure_wipe.h, 1259
rtklib_solution.h, 1260, 1262
Rtklib_Solver, 914
beidou_cnav1_ephemeris_map, 918
beidou_cnav1_iono, 918
beidou_cnav1_page_data_map, 918
beidou_cnav1_utc_model, 918
beidou_cnav2_ephemeris_map, 918
beidou_dnav_almanac_map, 918
beidou_dnav_ephemeris_map, 918
beidou_dnav_iono, 918
beidou_dnav_utc_model, 918
galileo_almanac_map, 919
galileo_ephemeris_map, 919
galileo_ephemeris_store, 919
galileo_iono, 919
galileo_reduced_ced_map, 919
galileo_utc_model, 919
get_broadcast_signal_health, 917
get_gdop, 917
get_hdop, 917
get_pdop, 917
get_vdop, 917
glonass_gnav_almanac, 919
glonass_gnav_almanac_map, 919
glonass_gnav_ephemeris_map, 919
glonass_gnav_utc_model, 919
gps_almanac_map, 919
gps_cnav_eop, 920
gps_cnav_ephemeris_map, 920
gps_cnav_iono, 920
gps_cnav_utc_model, 920
gps_ephemeris_map, 920
gps_iono, 920
gps_previous_ephemeris_map, 920
gps_utc_model, 920
has_precise_ephemeris, 917
NTRIP_MIN_COMMON_SATELLITES, 920
pvt_sol, 920
pvt_ssat, 920
qzss_cnav_eop, 921
qzss_cnav_iono, 921
qzss_cnav_utc_model, 921
qzss_iono, 921
qzss_utc_model, 921
solution_attempted, 918
rtklib_solver.h, 1364, 1365
rtklib_stream.h, 1264, 1268
FTP_CMD, 1266
FTP_TIMEOUT, 1266
MAXCLI, 1266
MAXSTATMSG, 1266
NTRIP_AGENT, 1266
NTRIP_CLI_PORT, 1266
NTRIP_MAXRSP, 1266
NTRIP_MAXSTR, 1266
NTRIP_RSP_ERROR, 1266
NTRIP_RSP_HTTP, 1267
NTRIP_RSP_OK_CLI, 1267
NTRIP_RSP_OK_SVR, 1267
NTRIP_RSP_SRCTBL, 1267
NTRIP_RSP_TBLEND, 1267
NTRIP_SVR_PORT, 1267
NTRIP_VERSION_1, 1267
NTRIP_VERSION_2, 1267
SERIBUFFSIZE, 1267
TIMETAGH_LEN, 1267
TINTACT, 1267
VER_RTKLIB, 1267
rtklib_tides.h, 1270, 1271
GME, 1271
GMM, 1271
GMS, 1271
rtklib_tls.h, 1272, 1273
Rtklib_Tls_Client, 921
Rtklib_Tls_Client_Test_Access, 922
Rtklib_Tls_Client_Test_Access
Rtklib_Tls_Client, 922
rtksvr_t, 922
buff, 922
buffsize, 922
cputime, 923
cycle, 923
files, 923
format, 923
ftime, 923
lock, 923
moni, 923
nav, 923
navsel, 923
nb, 923
nmeacycle, 923
nmeapos, 923
nmeareq, 924
nmsg, 924
npb, 924
nsb, 924
nsbs, 924
nsol, 924
obs, 924

pbuf, 924
prcout, 924
raw, 924
rtcm, 924
rtk, 924
sbsmsg, 925
sbuf, 925
solbuf, 925
solopt, 925
state, 925
stream, 925
thread, 925
tick, 925
RTL_TCP_COMMAND
signal_source_libs, 310
rtl_tcp_command
signal_source_libs, 310
rtl_tcp_commands.h, 1517, 1518
Rtl_Tcp_Dongle_Info, 925
get_tuner_gain_count, 926
get_tuner_type, 926
rtl_tcp_dongle_info.h, 1518, 1519
rtl_tcp_signal_source.h, 1446, 1447
rtl_tcp_signal_source_c, 926
rtl_tcp_signal_source_c.h, 1472, 1473
rtl_tcp_signal_source_c_sptr
signal_source_gr_blocks, 308
RtlTcpSignalSource, 927
connect, 928
disconnect, 928
get_left_block, 928
get_right_block, 928
item_size, 928
ru
prcopt_t, 849
run
ControlThread, 429
run_acquisition
Fpga_Acquisition, 509
run_server
Rtcm, 899
runner_up_margin
HistogramBitSynchronizer::Config, 424
running
GNSSFlowgraph, 643
Ntrip_Rtcm_Client_Status, 781
RURAI
Beidou_Dnav_Differential_Correction, 381
Rx
rtklib_rtkcmn.h, 1244
RX_time
Gnss_Synchro, 636
Monitor_Pvt, 751
rx_time
GnssTime, 645
Ry
rtklib_rtkcmn.h, 1244
Rz
rtklib_rtkcmn.h, 1245

saddr
tcp_t, 994
sample_counter
GalileoTowMapEntry, 574
TrackingCmd, 1012
sample_counter_delta
galileo_tow_utils.h, 1544
gnss_tow, 325
sample_freq_in
direct_resampler_conditioner_cb, 440
direct_resampler_conditioner_cc, 441
direct_resampler_conditioner_cs, 442
sample_freq_out
direct_resampler_conditioner_cb, 440
direct_resampler_conditioner_cc, 441
direct_resampler_conditioner_cs, 442
sample_offset
SensorDataFileConfiguration, 948
sample_period
SensorDataFileConfiguration, 948
sample_stamp
TOW_to_trk, 1007
sample_stamp_s
Sbas_Raw_Message, 934
sampled_ms
Acq_Conf, 335
samples
FileSourceBase, 494
GSS6450FileSignalSource, 695
MultichannelFileSignalSource, 757
samples_per_chip
Acq_Conf, 335
samples_per_code
Acq_Conf, 335
samples_per_ms
Acq_Conf, 335
samplestamp
PpsSamplestamp, 844
samplesToSkip
FileSourceBase, 494
sampling_frequency
FileSourceBase, 494
GSS6450FileSignalSource, 695
MultichannelFileSignalSource, 758
sat
alm_t, 353
eph_t, 474
geph_t, 579
half_cyc_tag, 697
lexeph_t, 734
obsd_t, 790
pcv_t, 837
sbssat_t, 941
sbssatp_t, 941
seph_t, 951
solstat_t, 977
stec_t, 987

sat_type
Bds3_B1c_AlmanacMedium, 360
Bds3_B1c_AlmanacReduced, 361
Beidou_Cnav1_Ephemeris, 373
satclass
tled_t, 1004
satClkDrift
Gnss_Ephemeris, 616
satellite_corrections
Sbas_Rtklib_Corrections, 935
satellite_mask
Galileo_HAS_data, 552
satellite_position
Glonass_Gnav_Almanac, 581
satellite_submask
Galileo_HAS_data, 552
satellite_visibility.h, 1715, 1716
satelliteBlock
Beidou_Dnav_Ephemeris, 387
Gps_Ephemeris, 667
satellitePosition
Gnss_Ephemeris, 613
satellitePosVelComputation
Gnss_Almanac, 604
SatelliteVisibility, 928
enabled, 929
search_ratio, 929
SetCommandReference, 929
Tick, 929
sateph
prcopt_t, 849
SatH2
Beidou_Dnav_Differential_Corrections, 382
satno
tled_t, 1004
satpos_X
Gnss_Ephemeris, 616
satpos_Y
Gnss_Ephemeris, 616
satpos_Z
Gnss_Ephemeris, 616
sats
msm_h_t, 754
satvel_X
Gnss_Ephemeris, 617
satvel_Y
Gnss_Ephemeris, 617
satvel_Z
Gnss_Ephemeris, 617
save_cnav_ephemeris_map_xml
Gnss_Sdr_Supl_Client, 625
save_cnav_utc_xml
Gnss_Sdr_Supl_Client, 625
save_ephemeris_map_xml
Gnss_Sdr_Supl_Client, 625
save_gal_almanac_xml
Gnss_Sdr_Supl_Client, 625
save_gal_ephemeris_map_xml
Gnss_Sdr_Supl_Client, 625
save_gal_iono_xml
Gnss_Sdr_Supl_Client, 625
save_gal_utc_xml
Gnss_Sdr_Supl_Client, 625
save_glo_utc_xml
Gnss_Sdr_Supl_Client, 626
save_gnav_ephemeris_map_xml
Gnss_Sdr_Supl_Client, 626
save_gps_almanac_xml
Gnss_Sdr_Supl_Client, 626
save_iono_xml
Gnss_Sdr_Supl_Client, 626
save_ref_location_xml
Gnss_Sdr_Supl_Client, 626
save_ref_time_xml
Gnss_Sdr_Supl_Client, 626
save_utc_xml
Gnss_Sdr_Supl_Client, 626
Sbas_Ephemeris, 930
b_sv_do_not_use, 930
d_acc, 930
d_af0, 931
d_af1, 931
d_pos, 931
d_tof, 931
d_vel, 931
i_prn, 931
i_sv_ura, 931
i_t0, 931
sbas_ephemeris.h, 2058, 2059
SBAS_L1.h, 2060, 2061
SBAS_L1_CHANNEL_SYMBOL_PERIOD_MS
core_system_parameters, 216
SBAS_L1_CHANNEL_SYMBOL_RATE_SPS
core_system_parameters, 216
sbas_l1_code_gen_complex
algorithms_libs, 253
sbas_l1_code_gen_complex_sampled
algorithms_libs, 254
sbas_l1_code_gen_float
algorithms_libs, 254
SBAS_L1_CODE_LENGTH_CHIPS
core_system_parameters, 216
SBAS_L1_CODE_PERIOD_MS
core_system_parameters, 216
SBAS_L1_CODE_PERIOD_S
core_system_parameters, 216
SBAS_L1_CODE_RATE_CPS
core_system_parameters, 216
SBAS_L1_DATA_FIELD_BYTES
core_system_parameters, 216
SBAS_L1_DATA_RATE_BPS
core_system_parameters, 216
SBAS_L1_FREQ_HZ
core_system_parameters, 217
SBAS_L1_MAX_COHERENT_INTEGRATION_SYMBOLS
core_system_parameters, 217

SBAS_L1_MSG_LENGTH_BITS
core_system_parameters, 217
SBAS_L1_MSG_LENGTH_BYTES
core_system_parameters, 217
SBAS_L1_OPT_ACQ_FS_SPS
core_system_parameters, 217
SBAS_L1_PREAMBLE_1
core_system_parameters, 217
SBAS_L1_PREAMBLE_2
core_system_parameters, 217
SBAS_L1_PREAMBLE_3
core_system_parameters, 217
SBAS_L1_PRN_MAX
core_system_parameters, 217
SBAS_L1_PRN_MIN
core_system_parameters, 217
sbas_l1_telemetry_decoder_gs, 932
consume_decoded_timestamps, 933
general_work, 933
reset, 933
set_channel, 933
set_satellite, 933
sbas_l1_telemetry_decoder_gs.h, 1538, 1539
sbas_l1_telemetry_decoder_gs_sptr
telemetry_decoder_gr_blocks, 312
Sbas_Raw_Message, 933
frame, 934
message_type, 934
prn, 934
sample_stamp_s, 934
Sbas_Raw_Message, 934
sbas_raw_message.h, 2062
Sbas_Rtklib_Corrections, 934
active_prn, 935
ionosphere_corrections, 935
pending_messages, 935
satellite_corrections, 935
sbas_rtklib_corrections.h, 1370
sbas_signal_replica.h, 1274
sbascorr
prcopt_t, 850
SbasL1DumpSession, 935
open, 935
sbassatsel
prcopt_t, 850
sbs_t, 936
msgs, 936
n, 936
nmax, 936
sbsfcorr_t, 936
ai, 936
dt, 936
iodf, 936
prc, 937
rrc, 937
t0, 937
udre, 937
sbsigp_t, 937
delay, 937
give, 937
lat, 937
lon, 937
t0, 938
sbsigpband_t, 938
bite, 938
bits, 938
x, 938
y, 938
sbsion
nav_t, 765
sbsion_t, 938
igp, 939
iodi, 939
nigp, 939
sbslcorr_t, 939
daf0, 939
daf1, 939
dpos, 939
dvel, 939
iode, 939
t0, 940
sbsmsg
raw_t, 877
rtksvr_t, 925
sbsmsg_t, 940
msg, 940
prn, 940
tow, 940
week, 940
sbssat
nav_t, 765
sbssat_t, 940
iodp, 941
nsat, 941
sat, 941
tlat, 941
sbssatp_t, 941
fcorr, 941
lcorr, 941
sat, 941
sbuf
rtksvr_t, 925
SC2RAD
core_system_parameters, 218
sclkstab
prcopt_t, 850
sdr_gnss_packet_t, 942
galileo_satellites, 942
gps_satellites, 942
height, 942
latitude, 942
longitude, 942
microseconds, 942
nsvfix, 942
reserved, 942
status, 943

velocity, 943
sdr_gnss_packet_t::[struct].sats, 331
search_ratio
SatelliteVisibility, 929
sec
gtime_t, 696
seconds
Agnss_Ref_Time, 350
seconds_to_ms
galileo_tow_utils.h, 1544
gnss_tow, 325
secure_clear_ntrip_credentials
pvt_libs, 302
Secure_String, 943
∼Secure_String, 944
clear, 944
empty, 944
operator=, 944
Secure_String, 943
value, 944
secure_string.h, 1371
send_gga
Ntrip_Rtcm_Client_Config, 780
send_message
Rtcm, 899
send_telemetry_msg
GNSSFlowgraph, 643
sensor_data_aggregator.h, 1275
sensor_data_file.h, 1276, 1277
sensor_data_resampler.h, 1278
sensor_data_source.h, 1279
sensor_data_source_configuration.h, 1280
SENSOR_DATA_TAG
SensorDataAggregator, 945
sensor_data_type.h, 1282
sensor_identifier.h, 1283
SensorDataAggregator, 945
SENSOR_DATA_TAG, 945
SensorDataConfiguration, 946
file_id, 946
id, 946
identifier, 946
offset, 946
tag_key, 946
type, 946
SensorDataFile, 946
id_type, 947
sptr, 947
SensorDataFileConfiguration, 947
chunk_size, 947
file_offset, 947
filename, 947
id, 947
repeat, 948
sample_offset, 948
sample_period, 948
SensorDataSample
SensorDataSample< DataType >, 948
SensorDataSample< DataType >, 948
SensorDataSample, 948
timestamp, 948
value, 948
SensorDataSource, 949
sptr, 949
SensorDataSourceConfiguration, 949
SensorDataType, 950
value_type, 950
SensorIdentifier, 950
value_type, 951
sep
solopt_t, 975
seph
nav_t, 765
seph_t, 951
acc, 951
af0, 951
af1, 951
pos, 951
sat, 951
sva, 951
svh, 951
t0, 952
tof, 952
vel, 952
seqno
rtcm_t, 904
sequence1
Mack_lookup, 739
sequence2
Mack_lookup, 739
Serdes_Galileo_Eph, 952
createProtobuffer, 953
operator=, 953
readProtobuffer, 953
Serdes_Galileo_Eph, 952
serdes_galileo_eph.h, 1373
Serdes_Gnss_Synchro, 953
createProtobuffer, 954
operator=, 954
readProtobuffer, 955
Serdes_Gnss_Synchro, 954
serdes_gnss_synchro.h, 1694, 1695
Serdes_Gps_Eph, 955
createProtobuffer, 956
operator=, 956
readProtobuffer, 956
Serdes_Gps_Eph, 955, 956
serdes_gps_eph.h, 1376
Serdes_Monitor_Pvt, 957
createProtobuffer, 957
operator=, 957, 958
readProtobuffer, 958
Serdes_Monitor_Pvt, 957
serdes_monitor_pvt.h, 1378, 1379
Serdes_Nav_Message, 958
createProtobuffer, 959

operator=, 959
readProtobuffer, 959
Serdes_Nav_Message, 958
serdes_nav_message.h, 1687
serial_t, 959
dev, 960
error, 960
serialize
Agnss_Ref_Location, 349
Agnss_Ref_Time, 350
Beidou_Cnav1_Ephemeris, 372
Beidou_Dnav_Almanac, 380
Beidou_Dnav_Ephemeris, 386
Beidou_Dnav_Iono, 389
Beidou_Dnav_Iono_Grid_Point, 390
Beidou_Dnav_Utc_Model, 396
Galileo_Almanac, 522
Galileo_Ephemeris, 536
Galileo_Iono, 559
Galileo_Utc_Model, 570
Glonass_Gnav_Almanac, 581
Glonass_Gnav_Ephemeris, 587
Glonass_Gnav_Utc_Model, 600
Gnss_Synchro, 633
Gps_Almanac, 649
Gps_CNAV_Ephemeris, 655
Gps_Ephemeris, 665
Gps_Iono, 669
Gps_Utc_Model, 680
Monitor_Pvt, 750
Monitor_Pvt::TrackedSatelliteInfo, 1008
SERIBUFFSIZE
algorithms_libs_rtklib, 292
rtklib_stream.h, 1267
server_name
Gnss_Sdr_Supl_Client, 628
server_port
Gnss_Sdr_Supl_Client, 628
set_active
acquisition_impl_interface, 340
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 531
galileo_pcps_8ms_acquisition_cc, 562
pcps_acquisition, 809
pcps_acquisition_fine_doppler_cc, 812
pcps_acquisition_fpga, 814
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 819
pcps_opencl_acquisition_cc, 822
pcps_quicksync_acquisition_cc, 824
pcps_tong_acquisition_cc, 827
set_alpha
Exponential_Smoother, 479
set_block_exp
Fpga_Acquisition, 509
set_channel
acquisition_impl_interface, 340
AcquisitionInterface, 342
beidou_b1c_telemetry_decoder_gs, 368
beidou_b2a_telemetry_decoder_gs, 369
beidou_dnav_telemetry_decoder_gs, 395
dll_pll_veml_tracking_fpga, 459
DllPllTrackingAdapter, 462
DllPllTrackingAdapterFpga, 464
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 531
galileo_pcps_8ms_acquisition_cc, 563
galileo_telemetry_decoder_gs, 568
GalileoE1TcpConnectorTracking, 573
glonass_gnav_telemetry_decoder_gs, 598
gps_l1_ca_telemetry_decoder_gs, 673
gps_l2c_telemetry_decoder_gs, 675
gps_l5_telemetry_decoder_gs, 676
Gps_Navigation_Message, 678
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
pcps_acquisition, 809
pcps_acquisition_fine_doppler_cc, 812
pcps_acquisition_fpga, 815
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 820
pcps_opencl_acquisition_cc, 822
pcps_quicksync_acquisition_cc, 825
pcps_tong_acquisition_cc, 827
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
sbas_l1_telemetry_decoder_gs, 933
telemetry_impl_interface, 999
TelemetryDecoderAdapter, 1000
Tlm_CRC_Stats, 1006
TrackingInterface, 1013
set_channel_fsm
acquisition_impl_interface, 340
AcquisitionInterface, 342
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 532
galileo_pcps_8ms_acquisition_cc, 563
pcps_acquisition, 810
pcps_acquisition_fine_doppler_cc, 812
pcps_acquisition_fpga, 815
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 820
pcps_opencl_acquisition_cc, 822
pcps_quicksync_acquisition_cc, 825
pcps_tong_acquisition_cc, 827
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
set_clock_drift_ppm
Pvt_Solution, 865
set_configuration
GNSSFlowgraph, 643
set_control_queue

ControlThread, 429
set_course_over_ground
Pvt_Solution, 865
set_DLL_BW
Tracking_2nd_DLL_filter, 1009
set_doppler_center
AcquisitionInterface, 342
pcps_acquisition, 810
pcps_acquisition_fpga, 815
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterFpga, 836
set_doppler_num_bins
AcquisitionInterface, 342
pcps_acquisition, 810
PcpsAcquisitionAdapter, 830
set_doppler_sweep
Fpga_Acquisition, 509
set_doppler_wipeoffs
CudaPcpsEngine, 436
set_ephemeris_data
OSNMA_NavData, 802
set_fft_codes
CudaPcpsEngine, 436
set_flag_ephemeris_str_1
Glonass_Gnav_Navigation_Message, 595
set_flag_ephemeris_str_2
Glonass_Gnav_Navigation_Message, 596
set_flag_ephemeris_str_3
Glonass_Gnav_Navigation_Message, 596
set_flag_ephemeris_str_4
Glonass_Gnav_Navigation_Message, 596
set_flag_new_SOW_available
Beidou_Dnav_Navigation_Message, 393
set_flag_TOW_new
Glonass_Gnav_Navigation_Message, 596
set_flag_TOW_set
Galileo_Fnav_Message, 541
Galileo_Inav_Message, 557
Glonass_Gnav_Navigation_Message, 596
set_flag_update_slot_number
Glonass_Gnav_Navigation_Message, 596
set_gnss_synchro
acquisition_impl_interface, 340
AcquisitionInterface, 342
dll_pll_veml_tracking_fpga, 459
DllPllTrackingAdapter, 462
DllPllTrackingAdapterFpga, 464
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 532
galileo_pcps_8ms_acquisition_cc, 563
GalileoE1TcpConnectorTracking, 573
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
pcps_acquisition, 810
pcps_acquisition_fine_doppler_cc, 813
pcps_acquisition_fpga, 815
pcps_assisted_acquisition_cc, 817
pcps_cccwsr_acquisition_cc, 820
pcps_opencl_acquisition_cc, 823
pcps_quicksync_acquisition_cc, 825
pcps_tong_acquisition_cc, 827
PcpsAcquisitionAdapter, 830
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
TrackingInterface, 1013
set_initial_sample
Fpga_Multicorrelator_8sc, 515
set_IOD_nav
OSNMA_NavData, 802
set_last_received_TOW
OSNMA_NavData, 802
set_last_received_WN
OSNMA_NavData, 802
set_local_code
acquisition_impl_interface, 340, 341
AcquisitionInterface, 343
Fpga_Acquisition, 509
galileo_e5a_noncoherentIQ_acquisition_caf_cc, 532
galileo_pcps_8ms_acquisition_cc, 563
pcps_acquisition, 810
pcps_acquisition_fine_doppler_cc, 813
pcps_acquisition_fpga, 815
pcps_assisted_acquisition_cc, 818
pcps_cccwsr_acquisition_cc, 820
pcps_opencl_acquisition_cc, 823
pcps_quicksync_acquisition_cc, 825
pcps_tong_acquisition_cc, 828
PcpsAcquisitionAdapter, 831
PcpsAcquisitionAdapterCustom, 833
PcpsAcquisitionAdapterFpga, 836
set_local_code_and_taps
Fpga_Multicorrelator_8sc, 515
set_merkle_root
Gnss_Crypto, 609
osnma_msg_receiver, 799
set_num_valid_observations
Pvt_Solution, 865
set_output_vectors
Fpga_Multicorrelator_8sc, 515
set_pdi
Tracking_2nd_DLL_filter, 1010
Tracking_2nd_PLL_filter, 1010
set_PLL_BW
Tracking_2nd_PLL_filter, 1010
set_prn_d
OSNMA_NavData, 802
set_property
FileConfiguration, 488
InMemoryConfiguration, 714
set_public_key
Gnss_Crypto, 609
set_public_key_type
Gnss_Crypto, 610

set_ref_gps_week
Pvt_Solution, 866
set_resampler_latency
AcquisitionInterface, 343
PcpsAcquisitionAdapter, 831
PcpsAcquisitionAdapterCustom, 834
PcpsAcquisitionAdapterFpga, 836
set_rf_link
Glonass_Gnav_Navigation_Message, 596
set_rx_pos
Pvt_Solution, 866
set_rx_vel
Pvt_Solution, 866
set_samples_for_initialization
Exponential_Smoother, 479
set_satellite
beidou_b1c_telemetry_decoder_gs, 368
beidou_b2a_telemetry_decoder_gs, 369
beidou_dnav_telemetry_decoder_gs, 395
galileo_telemetry_decoder_gs, 568
glonass_gnav_telemetry_decoder_gs, 598
gps_l1_ca_telemetry_decoder_gs, 673
gps_l2c_telemetry_decoder_gs, 675
gps_l5_telemetry_decoder_gs, 676
sbas_l1_telemetry_decoder_gs, 933
telemetry_impl_interface, 999
TelemetryDecoderAdapter, 1000
set_satellite_PRN
Beidou_Dnav_Navigation_Message, 393
Gps_Navigation_Message, 679
set_secondary_code_lengths
Fpga_Multicorrelator_8sc, 515
set_signal
Channel, 406
ChannelInterface, 411
set_signal_type
Beidou_Dnav_Navigation_Message, 393
set_speed_over_ground
Pvt_Solution, 866
set_switch_position
Fpga_Switch, 517
set_system
Gps_Almanac, 649
Gps_Ephemeris, 666
set_time_offset_s
Pvt_Solution, 866
set_time_stamp
Galileo_Cnav_Message, 529
set_tow
Galileo_Cnav_Message, 529
set_TOW0_flag
Galileo_Inav_Message, 557
set_TOW1_flag
Galileo_Fnav_Message, 542
set_TOW2_flag
Galileo_Fnav_Message, 542
set_TOW3_flag
Galileo_Fnav_Message, 542
set_TOW4_flag
Galileo_Fnav_Message, 542
set_TOW5_flag
Galileo_Inav_Message, 557
set_TOW6_flag
Galileo_Inav_Message, 557
set_tow_sf0
OSNMA_NavData, 803
set_update_verified_bits
OSNMA_NavData, 803
set_utc_data
OSNMA_NavData, 803
set_verified_status
OSNMA_NavData, 803
set_week
Galileo_Cnav_Message, 529
set_wn_sf0
OSNMA_NavData, 803
SetCommandReference
SatelliteVisibility, 929
SGN_LAMBDA
rtklib_lambda.h, 1217
shift_padding
IONGSMSChunkUnpackingCtx< WT >, 717
short_x2_to_cshort, 960
short_x2_to_cshort.h, 1284, 1285
short_x2_to_cshort_sptr
algorithms_libs, 238
show_local_time_zone
Pvt_Conf, 862
sif
Bds3_B1c_PageCommon, 363
sig_type
Beidou_Cnav1_Ephemeris, 373
Beidou_Dnav_Ephemeris, 387
SIGN
ion_gsms_stream_encodings.h, 1512
Signal
Gnss_Synchro, 637
signal
Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
HAS_obs_corrections, 699
Kf_Conf, 728
Monitor_Pvt::TrackedSatelliteInfo, 1008
Nav_Message_Packet, 760
Ntrip_Rtk_Signal, 784
TOW_to_trk, 1007
Signal Conditioner, 228
Signal model, 23
Signal Source, 304
signal_conditioner.h, 1090
Signal_Enabled_Flags, 960
check_any_enabled, 961
check_only_enabled, 961
flags, 961
has_beidou, 961
has_galileo, 961

has_glonass, 962
has_gps, 962
has_qzss, 962
only_beidou, 962
only_galileo, 962
only_glonass, 962
only_gps, 962
only_qzss, 962
or_all, 961
signal_enabled_flags
Pvt_Conf, 862
signal_enabled_flags.h, 1381, 1382
signal_flag
signal_flag.h, 2063
signal_flag.h, 2063, 2064
signal_flag, 2063
SIGNAL_FREQ_MAP
core_system_parameters, 218
signal_generator
GenSignalSource, 576
signal_generator.h, 1391
signal_generator_c, 962
signal_make_generator_c, 963
signal_generator_c.h, 1392, 1394
signal_generator_c_sptr, 1393
signal_make_generator_c, 1393
signal_generator_c_sptr
signal_generator_c.h, 1393
signal_health
Gps_CNAV_Ephemeris, 656
signal_make_generator_c
signal_generator_c, 963
signal_generator_c.h, 1393
signal_mask
Galileo_HAS_data, 552
signal_source_adapters, 304
signal_source_base.h, 1448
signal_source_gr_blocks, 306
ad936x_iio_source_sptr, 307
b_io_context, 307
bladerf_source_sptr, 308
Evk1029Source_sptr, 308
labsat23_source_sptr, 308
rtl_tcp_signal_source_c_sptr, 308
unpack_2bit_samples_sptr, 308
unpack_byte_2bit_cpx_samples_sptr, 308
unpack_byte_2bit_samples_sptr, 308
unpack_byte_4bit_samples_sptr, 308
unpack_gss6450_samples_sptr, 308
unpack_intspir_1bit_samples_sptr, 308
unpack_ntlab_2bit_samples_sptr, 308
unpack_short_byte_samples_sptr, 308
signal_source_interface.h, 1644
signal_source_libs, 309
FIR_BUF_SIZE, 310
iodev, 310
RTL_TCP_COMMAND, 310
rtl_tcp_command, 310
SignalConditioner, 963
∼SignalConditioner, 964
connect, 964
data_type_adapter, 964
disconnect, 965
get_left_block, 965
get_right_block, 965
implementation, 965
input_filter, 965
item_size, 965
resampler, 965
role, 965
SignalConditioner, 964
SignalGenerator, 965
connect, 966
disconnect, 966
get_left_block, 966
get_right_block, 966
implementation, 966
item_size, 967
role, 967
SignalSourceBase, 967
decode_item_type, 968
get_left_block, 969
getRfChannels, 969
implementation, 969
role, 969
SignalSourceBase, 968
SignalSourceInterface, 969
GNSS block interfaces, 84
sigs
msm_h_t, 754
SinBocCorrelationFunction
tracking_libs, 322
SISA
Galileo_Ephemeris, 538
sisaioc
Bds3_B1c_PageData, 364
sisaioe
Bds3_B1c_PageData, 364
sismai
Bds3_B1c_PageCommon, 363
size
algorithms_libs, 254
Concurrent_Map< Data >, 420
Concurrent_Queue< Data >, 422
SIZE_DSM_BLOCKS_BYTES
core_system_parameters, 218
Skew_symmetric
algorithms_libs, 254
skipped
Tag, 992
slip
ssat_t, 982
slipc
solstat_t, 977
ssat_t, 982
slope

Dll_Pll_Conf, 449
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
SM
ion_gsms_stream_encodings.h, 1512
SMA
ion_gsms_stream_encodings.h, 1512
smooth
msm_h_t, 754
smoother_length
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
smoothing_factor
Obs_Conf, 788
SNR
obsd_t, 790
snr
lexmsg_t, 735
solstat_t, 977
ssat_t, 982
snr_base
ssat_t, 982
snr_rover
ssat_t, 982
snrmask
prcopt_t, 850
snrmask_t, 971
ena, 971
mask, 971
sock
tcp_t, 994
socket_t
algorithms_libs_rtklib, 275
sol
rtk_t, 906
sol_t, 971
age, 971
dtr, 971
ns, 971
prev_ratio1, 972
prev_ratio2, 972
qr, 972
ratio, 972
rr, 972
stat, 972
thres, 972
time, 972
type, 972
solbuf
rtksvr_t, 925
solbuf_t, 972
buff, 973
cyclic, 973
data, 973
end, 973
n, 973
nb, 973
nmax, 973
rb, 973
start, 973
time, 973
SOLF_ENU
algorithms_libs_rtklib, 292
SOLF_GSIF
algorithms_libs_rtklib, 292
SOLF_LLH
algorithms_libs_rtklib, 292
SOLF_NMEA
algorithms_libs_rtklib, 292
SOLF_STAT
algorithms_libs_rtklib, 292
SOLF_XYZ
algorithms_libs_rtklib, 292
solopt
rtksvr_t, 925
SOLOPT_DEFAULT
rtklib_rtksvr.h, 1254
solopt_t, 974
datum, 974
degf, 974
geoid, 974
height, 974
maxsolstd, 974
nmeaintv, 974
outhead, 975
outopt, 975
posf, 975
prog, 975
sep, 975
solstatic, 975
sstat, 975
timef, 975
times, 975
timeu, 975
trace, 975
SOLQ_DGPS
algorithms_libs_rtklib, 292
SOLQ_DR
algorithms_libs_rtklib, 292
SOLQ_FIX
algorithms_libs_rtklib, 293
SOLQ_FLOAT
algorithms_libs_rtklib, 293
SOLQ_NONE
algorithms_libs_rtklib, 293
SOLQ_PPP
algorithms_libs_rtklib, 293
SOLQ_SBAS
algorithms_libs_rtklib, 293
SOLQ_SINGLE
algorithms_libs_rtklib, 293
solstat_t, 976
az, 976
el, 976
flag, 976

frq, 976
lock, 976
outc, 976
rejc, 976
resc, 976
resp, 977
sat, 977
slipc, 977
snr, 977
time, 977
solstatbuf_t, 977
data, 977
n, 977
nmax, 977
solstatic
solopt_t, 975
soltype
prcopt_t, 850
solution_attempted
Rtklib_Solver, 918
solution_status
Monitor_Pvt, 751
solution_type
Monitor_Pvt, 752
source
FileSourceBase, 494
FileTimestampSignalSource, 497
FourBitCpxFileSignalSource, 507
NsrFileSignalSource, 774
NTLabFileSignalSource, 777
Osnma_Merkle_Tree_Material, 797
Osnma_Public_Key_Material, 805
SpirFileSignalSource, 980
TwoBitCpxFileSignalSource, 1017
TwoBitPackedFileSignalSource, 1020
source_item_size
FileSourceBase, 495
source_PRN
Beidou_Dnav_Differential_Corrections, 382
SOW
Beidou_Dnav_Differential_Corrections, 382
spare1
Beidou_Dnav_Ephemeris, 387
Gps_Ephemeris, 667
spare2
Beidou_Dnav_Ephemeris, 387
Gps_Ephemeris, 667
spc
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
speed
file_t, 485
SPEED_OF_LIGHT_M_MS
core_system_parameters, 218
SPEED_OF_LIGHT_M_S
core_system_parameters, 218
SPI_close
Fpga_spidev, 516
SPI_open
Fpga_spidev, 516
spir_file_signal_source.h, 1449, 1450
SpirFileSignalSource, 978
create_file_source_hook, 980
itemTypeToSize, 980
post_disconnect_hook, 980
pre_connect_hook, 980
source, 980
sptr
FifoReader, 482
Gr_Complex_Ip_Packet_Source, 693
IONGSMSFileSource, 719
SensorDataFile, 947
SensorDataSource, 949
SQRT2
rtklib_ppp.h, 1226
SQRT_A_LSB
core_system_parameters, 218
sqrt_a_m_sqrt
Bds3_B1c_AlmanacMedium, 360
sqrtA
Gnss_Almanac, 606
Gnss_Ephemeris, 617
ssat
rtk_t, 906
ssat_t, 980
azel, 981
code, 981
fix, 981
gf, 981
gf2, 981
half, 981
lock, 981
mw, 981
outc, 982
ph, 982
phw, 982
pt, 982
rejc, 982
resc, 982
resp, 982
slip, 982
slipc, 982
snr, 982
snr_base, 982
snr_rover, 982
sys, 983
vs, 983
vsat, 983
ssr
nav_t, 766
rtcm_t, 904
ssr_t, 983
cbias, 983
dclk, 983
ddeph, 983

deph, 984
hrclk, 984
iod, 984
iodcrc, 984
iode, 984
pbias, 984
refd, 984
stdpb, 984
t0, 984
udi, 984
update, 984
ura, 984
yaw_ang, 985
yaw_rate, 985
SSRUDINT
rtklib_rtcm3.h, 1237
sstat
solopt_t, 975
sta
raw_t, 877
rtcm_t, 904
sta_t, 985
antdes, 985
antsetup, 985
antsno, 985
del, 985
deltype, 986
hgt, 986
itrf, 986
marker, 986
name, 986
pos, 986
recsno, 986
rectype, 986
recver, 986
stable_best_required
HistogramBitSynchronizer::Config, 424
stah
rtcm_t, 904
staid
rtcm_t, 904
start
ad936x_iio_source, 345
DMASignalSourceFPGA, 466
FifoReader, 482
file_t, 485
GNSSBlockInterface, 640
GNSSFlowgraph, 643
solbuf_t, 973
start_acquisition
Channel, 406
ChannelInterface, 411
start_tag_accumulation
OSNMA_NavData, 803
start_tracking
dll_pll_veml_tracking_fpga, 459
DllPllTrackingAdapter, 462
DllPllTrackingAdapterFpga, 464
GalileoE1TcpConnectorTracking, 573
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 689
TrackingInterface, 1013
stas
pppcorr_t, 843
stat
lexmsg_t, 736
sol_t, 972
state
ftp_t, 520
Ntrip_Rtcm_Client_Status, 781
ntrip_t, 786
Osnma_Merkle_Tree_Material, 797
Osnma_Public_Key_Material, 805
rtksvr_t, 925
stream_t, 989
tcp_t, 994
station_id
Ntrip_Rtcm_Client_Config, 780
Ntrip_Rtcm_Snapshot, 783
status
sdr_gnss_packet_t, 943
Tag, 992
std, 326
fcbd_t, 481
pclk_t, 807
peph_t, 839
prcopt_t, 850
stec_t, 987
trop_t, 1014
std::hash< SensorIdentifier::value_type >, 701
operator(), 701
stdpb
ssr_t, 984
stec
pppcorr_t, 843
stec_t, 986
azel, 987
flag, 987
ion, 987
sat, 987
std, 987
time, 987
STEP2_GRID
CudaPcpsEngine, 435
step_bytes
ad936x_iio_samples, 344
stop
ad936x_iio_source, 345
GNSSFlowgraph, 643
stop_acquisition
AcquisitionInterface, 343
Fpga_Acquisition, 509
pcps_acquisition_fpga, 815
PcpsAcquisitionAdapter, 831

PcpsAcquisitionAdapterCustom, 834
PcpsAcquisitionAdapterFpga, 836
stop_channel
Channel, 406
ChannelInterface, 411
stop_duplicated_satellite_channels
GNSSFlowgraph, 643
stop_server
Rtcm, 899
stop_tracking
dll_pll_veml_tracking_fpga, 459
DllPllTrackingAdapter, 462
DllPllTrackingAdapterFpga, 464
GalileoE1TcpConnectorTracking, 573
GpsL1CaDllPllTrackingGPU, 683
GpsL1CaGaussianTracking, 685
GpsL1CaKfTracking, 687
GpsL1CaTcpConnectorTracking, 690
TrackingInterface, 1013
store_public_key
Gnss_Crypto, 610
str
ntrip_t, 786
STR_FILE
algorithms_libs_rtklib, 275
STR_FTP
algorithms_libs_rtklib, 276
STR_HTTP
algorithms_libs_rtklib, 276
STR_MODE_R
algorithms_libs_rtklib, 276
STR_MODE_RW
algorithms_libs_rtklib, 276
STR_MODE_W
algorithms_libs_rtklib, 276
STR_NONE
algorithms_libs_rtklib, 276
STR_NTRIPCLI
algorithms_libs_rtklib, 276
STR_NTRIPSVR
algorithms_libs_rtklib, 276
STR_SERIAL
algorithms_libs_rtklib, 276
STR_TCPCLI
algorithms_libs_rtklib, 276
STR_TCPSVR
algorithms_libs_rtklib, 276
STR_UDP
algorithms_libs_rtklib, 276
stream
rtksvr_t, 925
stream_cfg, 987
bw_hz, 988
fs_hz, 988
lo_hz, 988
rfport, 988
stream_t, 988
inb, 988
inbt, 988
inr, 988
lock, 989
mode, 989
msg, 989
outb, 989
outbt, 989
outr, 989
path, 989
port, 989
state, 989
tact, 989
tick, 989
type, 989
StreamEncoding
ion_gsms_stream_encodings.h, 1510
STRFMT_NMEA
algorithms_libs_rtklib, 293
STRFMT_RNXCLK
algorithms_libs_rtklib, 293
STRFMT_RTCM2
algorithms_libs_rtklib, 293
STRFMT_RTCM3
algorithms_libs_rtklib, 293
STRFMT_SBAS
algorithms_libs_rtklib, 293
STRFMT_SP3
algorithms_libs_rtklib, 294
string_converter.h, 1689
string_decoder
Glonass_Gnav_Navigation_Message, 596
StringConverter, 990
subframe_decoder
Gps_Navigation_Message, 679
subfrm
raw_t, 877
SV_accuracy
Beidou_Dnav_Ephemeris, 387
Gps_Ephemeris, 667
sv_clock_drift
Glonass_Gnav_Ephemeris, 588
Gnss_Ephemeris, 613
SV_health
Beidou_Dnav_Almanac, 380
Beidou_Dnav_Ephemeris, 387
Gps_Almanac, 650
Gps_Ephemeris, 668
sva
eph_t, 474
geph_t, 579
seph_t, 951
svconf
alm_t, 353
svh
alm_t, 353
eph_t, 474
geph_t, 579
seph_t, 951

SVID1_7
Galileo_Almanac_Helper, 527
SVID2_8
Galileo_Almanac_Helper, 527
SVID3_9
Galileo_Almanac_Helper, 527
svr
tcpcli_t, 995
tcpsvr_t, 996
SWAP_D
rtklib_ppp.h, 1225
SWAP_I
rtklib_ppp.h, 1225
SWAP_LAMBDA
rtklib_lambda.h, 1218
swapintv
file_t, 485
swift_common.h, 1552
symbols
cnav_v27_part_t, 418
syncsol
prcopt_t, 850
sys
ssat_t, 983
SYS_ALL
algorithms_libs_rtklib, 294
SYS_BDS
algorithms_libs_rtklib, 294
SYS_GAL
algorithms_libs_rtklib, 294
SYS_GLO
algorithms_libs_rtklib, 294
SYS_GPS
algorithms_libs_rtklib, 294
SYS_IRN
algorithms_libs_rtklib, 294
SYS_LEO
algorithms_libs_rtklib, 294
SYS_NONE
algorithms_libs_rtklib, 294
SYS_QZS
algorithms_libs_rtklib, 295
SYS_SBS
algorithms_libs_rtklib, 295
sysefact
algorithms_libs_rtklib, 278
System
Gnss_Almanac, 606
Gnss_Ephemeris, 617
Gnss_Synchro, 637
system
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
Monitor_Pvt::TrackedSatelliteInfo, 1009
Nav_Message_Packet, 760
Ntrip_Rtk_Signal, 784
system_ecef_pos_sd_m
Pvt_Conf, 862
system_ecef_vel_sd_ms
Pvt_Conf, 862

t0
dgps_t, 439
lexion_t, 735
sbsfcorr_t, 937
sbsigp_t, 938
sbslcorr_t, 940
seph_t, 952
ssr_t, 984
t0_bgto_s
Bds3_B1c_Bgto, 362
t0a_7
Galileo_Almanac_Helper, 527
T0A_7_LSB
core_system_parameters, 218
t0a_9
Galileo_Almanac_Helper, 528
T0A_9_LSB
core_system_parameters, 218
T0C_4_LSB
core_system_parameters, 219
T0E_1_LSB
core_system_parameters, 219
T0T_6_LSB
core_system_parameters, 219
T_0_G_10_LSB
core_system_parameters, 219
t_0G
Galileo_Utc_Model, 571
t_eop
Gps_CNAV_Eop, 651
t_eop_s
Bds3_B1c_Eop, 363
T_GD_LSB
core_system_parameters, 219
T_OA
core_system_parameters, 135
T_OA_LSB
core_system_parameters, 219
T_OC_LSB
core_system_parameters, 219
T_OE_LSB
core_system_parameters, 219
T_OT_LSB
core_system_parameters, 219
tact
stream_t, 989
tcp_t, 994
Tag, 990
ADKD, 991
allowed_adkds, 991
computed_tag, 991
cop, 991
CTR, 992
e_verification_status, 991
id_counter, 992

nav_data, 992
nmas, 992
PRN_d, 992
PRNa, 992
received_tag, 992
skipped, 992
status, 992
Tag, 991
tag_id, 992
TOW, 992
WN, 992
tag
MACK_tag_and_info, 741
tag0
MACK_header, 738
tag0_valid
MACK_header, 738
tag_and_info
MACK_message, 740
tag_id
Tag, 992
tag_info
MACK_tag_and_info, 741
tag_key
SensorDataConfiguration, 946
taun
geph_t, 579
tbase
raw_t, 877
TC
ion_gsms_stream_encodings.h, 1512
TCA
ion_gsms_stream_encodings.h, 1512
tcon
tcp_t, 994
tcp
ntrip_t, 786
tcp_cmd_interface.h, 1719
Tcp_Communication, 993
tcp_communication.h, 1624, 1625
Tcp_Packet_Data, 993
proc_pack_carr_error, 993
proc_pack_carrier_doppler_hz, 993
proc_pack_code_error, 993
tcp_packet_data.h, 1626
tcp_t, 994
addr, 994
port, 994
saddr, 994
sock, 994
state, 994
tact, 994
tcon, 994
tdis, 995
tcpcli_t, 995
svr, 995
tirecon, 995
toinact, 995
TcpCmdInterface, 995
get_LLH, 996
get_utc_time, 996
tcpsvr_t, 996
cli, 996
svr, 996
tdis
tcp_t, 995
te
fcbd_t, 481
half_cyc_tag, 697
pcv_t, 837
tec
nav_t, 766
tec_t, 996
data, 997
hgts, 997
lats, 997
lons, 997
ndata, 997
rb, 997
rms, 997
time, 997
telemetry
Channel, 406
Telemetry_Decoder, 311
telemetry_decoder_adapter.h, 1520
Telemetry_Decoder_adapters, 311
telemetry_decoder_gr_blocks, 311
beidou_dnav_telemetry_decoder_gs_sptr, 312
galileo_telemetry_decoder_gs_sptr, 312
glonass_gnav_telemetry_decoder_gs_sptr, 312
gps_l1_ca_telemetry_decoder_gs_sptr, 312
gps_l2c_telemetry_decoder_gs_sptr, 312
gps_l5_telemetry_decoder_gs_sptr, 312
L1LnavSystem, 313
sbas_l1_telemetry_decoder_gs_sptr, 312
telemetry_impl_interface_sptr, 312
telemetry_decoder_interface.h, 1645
telemetry_decoder_libs, 315
telemetry_decoder_libswiftcnav, 313
ABS, 314
CLAMP_DIFF, 314
GPS_L2_V27_HISTORY_LENGTH_BITS, 314
GPS_L2C_V27_DECODE_BITS, 314
GPS_L2C_V27_DELAY_BITS, 314
GPS_L2C_V27_INIT_BITS, 314
MAX, 314
MIN, 314
telemetry_impl_interface, 997
reset, 999
set_channel, 999
set_satellite, 999
telemetry_impl_interface, 998
telemetry_impl_interface.h, 1541, 1542
telemetry_impl_interface_sptr
telemetry_decoder_gr_blocks, 312
TelemetryDecoderAdapter, 999

connect, 1000
disconnect, 1000
get_left_block, 1000
get_right_block, 1000
implementation, 1000
item_size, 1000
reset, 1000
role, 1000
set_channel, 1000
set_satellite, 1000
TelemetryDecoderInterface, 1001
temp_buffer_needed
cl_fft_plan, 413
tempmemobj
cl_fft_plan, 413
tempmemobj_imag
cl_fft_plan, 413
tempmemobj_real
cl_fft_plan, 413
tentative_events_required
HistogramBitSynchronizer::Config, 424
TEXT_BLACK
core_system_parameters, 219
TEXT_BLUE
core_system_parameters, 219
TEXT_BOLD_BLACK
core_system_parameters, 219
TEXT_BOLD_BLUE
core_system_parameters, 220
TEXT_BOLD_CYAN
core_system_parameters, 220
TEXT_BOLD_GREEN
core_system_parameters, 220
TEXT_BOLD_MAGENTA
core_system_parameters, 220
TEXT_BOLD_RED
core_system_parameters, 220
TEXT_BOLD_WHITE
core_system_parameters, 220
TEXT_BOLD_YELLOW
core_system_parameters, 220
TEXT_CYAN
core_system_parameters, 220
TEXT_GREEN
core_system_parameters, 220
TEXT_MAGENTA
core_system_parameters, 220
TEXT_RED
core_system_parameters, 220
TEXT_RESET
core_system_parameters, 220
TEXT_WHITE
core_system_parameters, 221
TEXT_YELLOW
core_system_parameters, 221
TGD
Gps_CNAV_Ephemeris, 656
Gps_Ephemeris, 668
tgd
eph_t, 474
lexeph_t, 734
TGD1
Beidou_Dnav_Ephemeris, 387
TGD2
Beidou_Dnav_Ephemeris, 388
TGD_B1Cp
Beidou_Cnav1_Ephemeris, 373
TGD_B2ap
Beidou_Cnav1_Ephemeris, 373
there_are_e1_channels
Tlm_Conf, 1005
there_are_e6_channels
Tlm_Conf, 1005
thread
ftp_t, 520
rtksvr_t, 925
three_bit_look_up
ion_gsms_stream_encodings.h, 1512
thres
sol_t, 972
THRES_RES
rtklib_ppp.h, 1226
thresar
prcopt_t, 850
thresdop
prcopt_t, 850
threshold
Acq_Conf, 335
Acq_Conf_Fpga, 338
thresslip
prcopt_t, 850
Tick
SatelliteVisibility, 929
tick
file_t, 485
rtksvr_t, 925
stream_t, 989
tick_f
file_t, 485
tidecorr
prcopt_t, 850
time
file_t, 485
gtime_t, 696
obsd_t, 790
pclk_t, 807
peph_t, 839
raw_t, 877
rtcm_t, 904
sol_t, 972
solbuf_t, 973
solstat_t, 977
stec_t, 987
tec_t, 997
trop_t, 1014
time_s

msm_h_t, 754
rtcm_t, 904
time_stamp
Galileo_HAS_page, 554
timed_wait_and_pop
Concurrent_Queue< Data >, 422
timef
solopt_t, 975
timeout_ms
Ntrip_Rtcm_Client_Config, 780
times
solopt_t, 975
TIMES_GPST
algorithms_libs_rtklib, 295
TIMES_JST
algorithms_libs_rtklib, 295
TIMES_UTC
algorithms_libs_rtklib, 295
timestamp
SensorDataSample< DataType >, 948
timetag
file_t, 485
TIMETAGH_LEN
algorithms_libs_rtklib, 295
rtklib_stream.h, 1267
timeu
solopt_t, 975
tint
url_t, 1026
tint_s
msm_h_t, 754
TINTACT
algorithms_libs_rtklib, 295
rtklib_stream.h, 1267
tirecon
tcpcli_t, 995
tlat
sbssat_t, 941
tle_t, 1001
data, 1002
n, 1002
nmax, 1002
tled_t, 1002
alias, 1002
bstar, 1002
desig, 1003
ecc, 1003
eleno, 1003
epoch, 1003
etype, 1003
inc, 1003
M, 1003
n, 1003
name, 1003
nddot, 1003
ndot, 1003
OMG, 1003
omg, 1004
rev, 1004
satclass, 1004
satno, 1004
Tlm_Conf, 1004
dump, 1004
dump_crc_stats, 1004
dump_crc_stats_filename, 1005
dump_filename, 1005
dump_mat, 1005
enable_navdata_monitor, 1005
enable_reed_solomon, 1005
remove_dat, 1005
there_are_e1_channels, 1005
there_are_e6_channels, 1005
tow_to_trk, 1005
use_ced, 1005
tlm_conf.h, 1553
Tlm_CRC_Stats, 1006
initialize, 1006
set_channel, 1006
update_CRC_stats, 1006
tlm_crc_stats.h, 1554
tlm_utils.h, 1555
tls_ctx
ntrip_t, 786
tls_enabled
Ntrip_Rtcm_Client_Config, 780
ntrip_t, 786
tnext
ftp_t, 520
to_ms
gnss_tow, 325
toa
alm_t, 353
Gnss_Almanac, 606
toa_s
Bds3_B1c_AlmanacMedium, 360
Bds3_B1c_PageData, 365
toas
alm_t, 353
tobs
raw_t, 878
toc
eph_t, 474
Gnss_Ephemeris, 617
tod
raw_t, 878
Todo List, 31
toe
eph_t, 474
geph_t, 579
Gnss_Ephemeris, 617
lexeph_t, 734
toe1
Gps_CNAV_Ephemeris, 657
toe2
Gps_CNAV_Ephemeris, 657
toes

eph_t, 474
tof
geph_t, 579
lexeph_t, 734
seph_t, 952
togeod
algorithms_libs, 254
toh
mt1_header, 756
toinact
tcpcli_t, 995
top
Gps_CNAV_Ephemeris, 657
topocent
algorithms_libs, 255
topts
ftp_t, 520
tot
Beidou_Cnav1_Utc_Model, 376
Galileo_Utc_Model, 571
Gps_Utc_Model, 681
total_block_exp
Acq_Conf_Fpga, 338
TOTRedCED
Galileo_Reduced_CED, 566
TOW
MACK_message, 740
Tag, 992
tow
Agnss_Ref_Time, 351
cnav_msg_t, 416
Galileo_HAS_data, 552
Galileo_HAS_page, 554
Galileo_Iono, 560
Gnss_Ephemeris, 617
Gps_Acq_Assist, 647
Gps_CNAV_Eop, 651
sbsmsg_t, 940
TOW_to_trk, 1007
TOW_at_current_symbol_ms
Gnss_Synchro, 637
Monitor_Pvt, 752
tow_at_current_symbol_ms
Nav_Message_Packet, 760
tow_ms
GalileoTowMapEntry, 574
GnssTime, 645
tow_ms_fraction
GnssTime, 645
TOW_sf0
OSNMA_msg, 798
TOW_to_trk, 1006
channel, 1007
prn, 1007
sample_stamp, 1007
signal, 1007
tow, 1007
TOW_to_trk, 1007
wn, 1007
tow_to_trk
Dll_Pll_Conf, 450
Tlm_Conf, 1005
tow_to_trk.h, 2064
tow_utils.h, 1556, 1557
towh_k
DSM_KROOT_message, 468
trace
solopt_t, 975
track_pilot
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 728
tracked_satellites
Monitor_Pvt, 752
Tracking, 315
tracking
Channel, 406
Tracking_2nd_DLL_filter, 1009
get_code_nco, 1009
initialize, 1009
set_DLL_BW, 1009
set_pdi, 1010
tracking_2nd_DLL_filter.h, 1627
Tracking_2nd_PLL_filter, 1010
set_pdi, 1010
set_PLL_BW, 1010
tracking_2nd_PLL_filter.h, 1628
tracking_adapters, 315
tracking_discriminators.h, 1629, 1630
Tracking_FLL_PLL_filter, 1010
tracking_FLL_PLL_filter.h, 1633
tracking_gr_blocks, 316
dll_pll_veml_tracking_fpga_sptr, 316
dll_pll_veml_tracking_sptr, 316
galileo_e1_tcp_connector_tracking_cc_sptr, 316
gps_l1_ca_dll_pll_tracking_gpu_cc_sptr, 317
gps_l1_ca_gaussian_tracking_cc_sptr, 317
gps_l1_ca_tcp_connector_tracking_cc_sptr, 317
tracking_interface.h, 1646, 1647
tracking_libs, 317
b_io_context, 319
CalculateSlope, 319
CalculateSlopeAbs, 319
carrier_lock_detector, 319
cn0_m2m4_estimator, 319
cn0_svn_estimator, 320
CosBocCorrelationFunction, 320
dll_nc_e_minus_l_normalized, 320
dll_nc_vemlp_normalized, 321
fll_four_quadrant_atan, 321
GetYIntercept, 321
GetYInterceptAbs, 321
NUM_RX_VARIABLES, 319
NUM_TX_VARIABLES_GALILEO_E1, 319
NUM_TX_VARIABLES_GPS_L1_CA, 319
phase_unwrap, 322

pll_cloop_two_quadrant_atan, 322
pll_four_quadrant_atan, 322
SinBocCorrelationFunction, 322
Tracking_loop_filter, 1011
operator=, 1011
Tracking_loop_filter, 1011
tracking_loop_filter.h, 1634
Tracking_sample_counter
Gnss_Synchro, 637
TrackingCmd, 1012
carrier_freq_hz, 1012
carrier_freq_rate_hz_s, 1012
code_freq_chips, 1012
enable_carrier_nco_cmd, 1012
enable_code_nco_cmd, 1012
sample_counter, 1012
trackingcmd.h, 1285, 1286
TrackingInterface, 1012
set_channel, 1013
set_gnss_synchro, 1013
start_tracking, 1013
stop_tracking, 1013
transition_confidence
HistogramBitSynchronizer::Config, 424
transition_window_epochs
HistogramBitSynchronizer::Config, 424
trop
pppcorr_t, 843
trop_t, 1014
std, 1014
time, 1014
trp, 1014
tropopt
prcopt_t, 850
TROPOPT_COR
algorithms_libs_rtklib, 295
TROPOPT_CORG
algorithms_libs_rtklib, 295
TROPOPT_EST
algorithms_libs_rtklib, 295
TROPOPT_ESTG
algorithms_libs_rtklib, 296
TROPOPT_OFF
algorithms_libs_rtklib, 296
TROPOPT_SAAS
algorithms_libs_rtklib, 296
TROPOPT_SBAS
algorithms_libs_rtklib, 296
trp
trop_t, 1014
truncated_week_diff
Galileo_Utc_Model, 570
try_pop
Concurrent_Queue< Data >, 422
ts
DSM_KROOT_message, 468
fcbd_t, 481
half_cyc_tag, 697
pcv_t, 837
tspan
lexion_t, 735
tt
rtk_t, 906
TTOL_MOVEB
algorithms_libs_rtklib, 296
ttr
eph_t, 474
ttt
lexmsg_t, 736
twist_kernel
cl_fft_plan, 413
two_bit_cpx_file_signal_source.h, 1450, 1451
two_bit_look_up
ion_gsms_stream_encodings.h, 1512
two_bit_packed_file_signal_source.h, 1452
TWO_N10
core_system_parameters, 221
TWO_N11
core_system_parameters, 221
TWO_N14
core_system_parameters, 221
TWO_N15
core_system_parameters, 221
TWO_N16
core_system_parameters, 221
TWO_N17
core_system_parameters, 221
TWO_N18
core_system_parameters, 221
TWO_N19
core_system_parameters, 221
TWO_N2
core_system_parameters, 222
TWO_N20
core_system_parameters, 222
TWO_N21
core_system_parameters, 222
TWO_N22
core_system_parameters, 222
TWO_N23
core_system_parameters, 222
TWO_N24
core_system_parameters, 222
TWO_N25
core_system_parameters, 222
TWO_N26
core_system_parameters, 222
TWO_N27
core_system_parameters, 222
TWO_N29
core_system_parameters, 222
TWO_N30
core_system_parameters, 223
TWO_N31
core_system_parameters, 223
TWO_N32

core_system_parameters, 223
TWO_N33
core_system_parameters, 223
TWO_N34
core_system_parameters, 223
TWO_N35
core_system_parameters, 223
TWO_N38
core_system_parameters, 223
TWO_N39
core_system_parameters, 223
TWO_N40
core_system_parameters, 223
TWO_N43
core_system_parameters, 223
TWO_N44
core_system_parameters, 224
TWO_N46
core_system_parameters, 224
TWO_N48
core_system_parameters, 224
TWO_N5
core_system_parameters, 224
TWO_N50
core_system_parameters, 224
TWO_N51
core_system_parameters, 224
TWO_N55
core_system_parameters, 224
TWO_N57
core_system_parameters, 224
TWO_N59
core_system_parameters, 224
TWO_N6
core_system_parameters, 224
TWO_N60
core_system_parameters, 225
TWO_N66
core_system_parameters, 225
TWO_N68
core_system_parameters, 225
TWO_N8
core_system_parameters, 225
TWO_N9
core_system_parameters, 225
TWO_P11
core_system_parameters, 225
TWO_P12
core_system_parameters, 225
TWO_P14
core_system_parameters, 225
TWO_P16
core_system_parameters, 225
TWO_P19
core_system_parameters, 225
TWO_P3
core_system_parameters, 226
TWO_P31
core_system_parameters, 226
TWO_P32
core_system_parameters, 226
TWO_P4
core_system_parameters, 226
TWO_P56
core_system_parameters, 226
TWO_P57
core_system_parameters, 226
TWO_P8
core_system_parameters, 226
TWO_PI
core_system_parameters, 226
TwoBitCpxFileSignalSource, 1014
create_file_source_hook, 1016
itemTypeToSize, 1016
packetsPerSample, 1016
pre_connect_hook, 1017
pre_disconnect_hook, 1017
source, 1017
TwoBitPackedFileSignalSource, 1017
create_file_source_hook, 1019
itemTypeToSize, 1019
packetsPerSample, 1019
pre_connect_hook, 1019
pre_disconnect_hook, 1019
source, 1020
type
lexmsg_t, 736
ntrip_t, 786
pcv_t, 837
SensorDataConfiguration, 946
sol_t, 972
stream_t, 989
url_t, 1026

udi
ssr_t, 984
udp_addresses
Pvt_Conf, 862
udp_eph_addresses
Pvt_Conf, 862
udp_eph_port
Pvt_Conf, 862
udp_ports
Pvt_Conf, 862
udre
dgps_t, 440
sbsfcorr_t, 937
UDREI
Beidou_Dnav_Differential_Correction, 381
uhd_signal_source.h, 1454
UhdSignalSource, 1020
connect, 1021
disconnect, 1021
get_left_block, 1021
get_right_block, 1021
item_size, 1021
uid

Osnma_Merkle_Tree_Material, 797
uio_fpga.h, 1690, 1691
uncertainty
Agnss_Ref_Location, 349
UNKNOWN
ion_gsms_stream_encodings.h, 1513
unlock_channel
Fpga_Multicorrelator_8sc, 515
unpack_2bit_samples, 1021
unpack_2bit_samples.h, 1475
unpack_2bit_samples_sptr
signal_source_gr_blocks, 308
unpack_byte_2bit_cpx_samples, 1022
unpack_byte_2bit_cpx_samples.h, 1477
unpack_byte_2bit_cpx_samples_sptr
signal_source_gr_blocks, 308
unpack_byte_2bit_samples, 1023
unpack_byte_2bit_samples.h, 1478, 1479
unpack_byte_2bit_samples_sptr
signal_source_gr_blocks, 308
unpack_byte_4bit_samples, 1023
unpack_byte_4bit_samples.h, 1480
unpack_byte_4bit_samples_sptr
signal_source_gr_blocks, 308
unpack_gss6450_samples, 1024
unpack_gss6450_samples.h, 1481
unpack_gss6450_samples_sptr
signal_source_gr_blocks, 308
unpack_intspir_1bit_samples, 1024
unpack_intspir_1bit_samples.h, 1482, 1483
unpack_intspir_1bit_samples_sptr
signal_source_gr_blocks, 308
unpack_ntlab_2bit_samples, 1025
unpack_ntlab_2bit_samples.h, 1484
unpack_ntlab_2bit_samples_sptr
signal_source_gr_blocks, 308
unpack_short_byte_samples, 1025
unpack_short_byte_samples.h, 1485
unpack_short_byte_samples_sptr
signal_source_gr_blocks, 308
UnscentedFilter, 1026
update
HistogramBitSynchronizer, 704
ssr_t, 984
update_CRC_stats
Tlm_CRC_Stats, 1006
update_local_code
Fpga_Multicorrelator_8sc, 515
update_PRN
Gnss_Satellite, 620
update_prn_code_length
Fpga_Multicorrelator_8sc, 515
UpdateVisibilityReference
GNSSFlowgraph, 643
URA
Gps_CNAV_Ephemeris, 657
ura
lexeph_t, 734
ssr_t, 984
URA0
Gps_CNAV_Ephemeris, 657
URA1
Gps_CNAV_Ephemeris, 657
URA2
Gps_CNAV_Ephemeris, 657
URAED
Gps_CNAV_Ephemeris, 657
URANED0
Gps_CNAV_Ephemeris, 658
URANED1
Gps_CNAV_Ephemeris, 658
URANED2
Gps_CNAV_Ephemeris, 658
url
ntrip_t, 786
url_t, 1026
dir, 1026
path, 1026
tint, 1026
type, 1026
use_automatic_resampler
Acq_Conf, 336
use_ced
Tlm_Conf, 1005
use_CFAR_algorithm_flag
Acq_Conf, 336
use_cuda
Acq_Conf, 336
use_e6_for_pvt
Pvt_Conf, 862
use_has_corrections
Pvt_Conf, 862
use_phase_dot_detector
HistogramBitSynchronizer::Config, 425
use_unhealthy_sats
Pvt_Conf, 862
used
Monitor_Pvt::TrackedSatelliteInfo, 1009
user
ftp_t, 520
ntrip_t, 787
user_clk_drift_ppm
Monitor_Pvt, 752
user_clk_offset
Monitor_Pvt, 752
username
Ntrip_Rtcm_Client_Config, 780
ut1_utc
erpd_t, 476
utc_cmp
nav_t, 766
utc_gal
nav_t, 766
utc_glo
nav_t, 766
utc_gps

nav_t, 766
utc_irn
nav_t, 766
utc_qzs
nav_t, 766
utc_sbs
nav_t, 766
utc_time
Beidou_Dnav_Navigation_Message, 393
Glonass_Gnav_Utc_Model, 600
Gps_Navigation_Message, 679
Monitor_Pvt, 752

v1_retry_requested
ntrip_t, 787
v27_decision_t, 1027
w, 1027
v27_poly_t, 1027
c0, 1027
c1, 1027
v27_t, 1027
decisions, 1028
decisions_count, 1028
decisions_index, 1028
metrics1, 1028
metrics2, 1028
new_metrics, 1028
old_metrics, 1028
poly, 1028
V27POLYA
fec.h, 1551
V27POLYB
fec.h, 1551
valid
Agnss_Ref_Location, 349
Agnss_Ref_Time, 351
Beidou_Cnav1_Iono, 374
Beidou_Cnav1_Utc_Model, 376
Beidou_Dnav_Utc_Model, 397
Galileo_Iono, 560
Glonass_Gnav_Utc_Model, 601
Gps_CNAV_Eop, 651
Gps_Iono, 670
Gps_Utc_Model, 681
half_cyc_tag, 697
MACK_tag_and_info, 741
Osnma_Merkle_Tree_Material, 797
Osnma_Public_Key_Material, 805
valid_sats
Monitor_Pvt, 752
valid_symbol
beidou_ldpc.h, 1776
valid_until
HAS_bias_corrections, 698
HAS_clock_corrections, 698
HAS_orbit_corrections, 700
validate
Ntrip_Rtcm_Client_Config, 779
validity_interval_index_clock_fullset_corrections
Galileo_HAS_data, 552
validity_interval_index_clock_subset_corrections
Galileo_HAS_data, 552
validity_interval_index_code_bias_corrections
Galileo_HAS_data, 552
validity_interval_index_orbit_corrections
Galileo_HAS_data, 552
validity_interval_index_phase_bias_corrections
Galileo_HAS_data, 553
value
Secure_String, 944
SensorDataSample< DataType >, 948
value_type
SensorDataType, 950
SensorIdentifier, 951
var
opt_t, 791
pcv_t, 837
VAR_ACC
algorithms_libs_rtklib, 296
VAR_BIAS
rtklib_ppp.h, 1226
VAR_CLK
rtklib_ppp.h, 1226
VAR_GRA
algorithms_libs_rtklib, 296
VAR_GRA_PPP
rtklib_ppp.h, 1226
VAR_HOLDAMB
algorithms_libs_rtklib, 296
VAR_HWBIAS
algorithms_libs_rtklib, 296
VAR_IONO_OFF
rtklib_ppp.h, 1226
VAR_NOTEC
rtklib_ionex.h, 1215
var_offsets
BeidouLdpcGraph, 398
VAR_POS
algorithms_libs_rtklib, 296
VAR_POS_PPP
rtklib_ppp.h, 1226
var_to_check
BeidouLdpcGraph, 398
var_to_edge
BeidouLdpcGraph, 399
var_to_h
BeidouLdpcGraph, 399
var_to_h_inv
BeidouLdpcGraph, 399
VAR_VEL
algorithms_libs_rtklib, 296
VAR_ZTD
rtklib_ppp.h, 1226
VARERR_MIN_EL
rtklib_pntpos.h, 1221
varholdamb
prcopt_t, 851

vco
peph_t, 839
vdop
Monitor_Pvt, 752
vector_length
Acq_Conf, 336
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 729
vel
geph_t, 579
lexeph_t, 734
peph_t, 839
seph_t, 952
vel_e
Monitor_Pvt, 752
vel_n
Monitor_Pvt, 752
vel_u
Monitor_Pvt, 752
vel_x
Monitor_Pvt, 752
vel_y
Monitor_Pvt, 753
vel_z
Monitor_Pvt, 753
velocity
sdr_gnss_packet_t, 943
VER_RTKLIB
algorithms_libs_rtklib, 277
rtklib_stream.h, 1267
verified
DSM_KROOT_message, 468
verify_dsm_pkr
osnma_msg_receiver, 800
verify_signature_ecdsa_p256
Gnss_Crypto, 610
verify_signature_ecdsa_p521
Gnss_Crypto, 610
version
Ntrip_Rtcm_Client_Config, 780
ntrip_t, 787
vertical_delay
Beidou_Dnav_Iono_Grid_Point, 390
very_early_late_space_chips
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 456
Kf_Conf, 729
very_early_late_space_narrow_chips
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 457
Kf_Conf, 729
visibility_aware_search_enabled
GNSSFlowgraph, 644
Viterbi_Decoder, 1028
decode, 1029
reset, 1029
Viterbi_Decoder, 1029
viterbi_decoder.h, 1559
Viterbi_Decoder_Sbas, 1030
decode_block, 1030
viterbi_decoder_sbas.h, 1561
vs
ssat_t, 983
vsat
ssat_t, 983
vst
peph_t, 839

w
v27_decision_t, 1027
wait
GNSSFlowgraph, 644
wait_and_pop
Concurrent_Queue< Data >, 422
week
Agnss_Ref_Time, 351
alm_t, 353
eph_t, 475
Galileo_HAS_data, 553
Galileo_HAS_page, 554
GalileoTowMapEntry, 574
GnssTime, 645
Monitor_Pvt, 753
raw_t, 878
sbsmsg_t, 940
week_after_delta
galileo_tow_utils.h, 1544
gnss_tow, 325
WEEK_MS
galileo_tow_utils.h, 1545
gnss_tow, 326
WEEK_S
gnss_tow, 326
WEEKOFFSET
rtklib_sbas.h, 1257
what_block
Gnss_Satellite, 620
with_word_type
ion_gsms_chunk_data.h, 1505
wlbias
nav_t, 766
WN
Galileo_Iono, 560
Galileo_Reduced_CED, 567
Gnss_Ephemeris, 618
MACK_message, 740
Tag, 992
wn
TOW_to_trk, 1007
wn0_bgto
Bds3_B1c_Bgto, 362
WN_0G
Galileo_Utc_Model, 571
WN_a_7
Galileo_Almanac_Helper, 528
WN_a_9

Galileo_Almanac_Helper, 528
wn_k
DSM_KROOT_message, 468
WN_LSF
Beidou_Cnav1_Utc_Model, 376
Beidou_Dnav_Utc_Model, 397
Galileo_Utc_Model, 571
Gps_Utc_Model, 681
WN_LSF_LSB
core_system_parameters, 226
WN_sf0
OSNMA_msg, 798
WN_T
Gps_Utc_Model, 681
WN_t
Beidou_Cnav1_Utc_Model, 377
WN_T_LSB
core_system_parameters, 226
WNa
Gnss_Almanac, 606
wna
Bds3_B1c_AlmanacMedium, 360
Bds3_B1c_PageData, 365
WNop
Gps_CNAV_Ephemeris, 658
WNot
Galileo_Utc_Model, 571
word
rtcm_t, 904
word_bitsize_
IONGSMSChunkUnpackingCtx< WT >, 718
word_count_
IONGSMSChunkUnpackingCtx< WT >, 718
work
rtklib_pvt_gs, 913
wrap_ms
galileo_tow_utils.h, 1544
gnss_tow, 325
wrap_s
gnss_tow, 325
wrap_s_to_ms
gnss_tow, 326
write
Concurrent_Map< Data >, 420
write_local_code
Fpga_Acquisition, 509
write_matlab_var
matlab_writter_helper.h, 1170
write_reg32
Fpga_spidev, 517
write_value
dump_logger_helper.h, 1138
wtime
file_t, 485

x
clFFT_Dim3, 414
rtk_t, 906
sbsigpband_t, 938
xa
rtk_t, 906
xml_output_enabled
Pvt_Conf, 863
xml_output_path
Pvt_Conf, 863
xml_path
Osnma_Merkle_Tree_Material, 797
xp
erpd_t, 476
xpr
erpd_t, 476

y
clFFT_Dim3, 414
sbsigpband_t, 938
y_intercept
Dll_Pll_Conf, 450
Dll_Pll_Conf_Fpga, 457
Kf_Conf, 729
yaw_ang
ssr_t, 985
yaw_rate
ssr_t, 985
yp
erpd_t, 476
ypr
erpd_t, 476

z
clFFT_Dim3, 414
zero_padding
Acq_Conf, 336
zmq_signal_source.h, 1455, 1456
ZmqSignalSource, 1030
connect, 1032
disconnect, 1032
get_right_block, 1032
item_size, 1032